

Telex M. N. Ngatched
Isaac Woungang
Jules-Raymond Tapamo
Serestina Viriri (Eds.)



632

Pan-African Artificial Intelligence and Smart Systems

LNICST

Third Pan-African Conference, PAAISS 2024
Durban, South Africa, December 4–6, 2024
Proceedings, Part II

Part 2



Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering

632

Editorial Board Members

Ozgur Akan, *Middle East Technical University, Ankara, Türkiye*
Paolo Bellavista, *University of Bologna, Bologna, Italy*
Jiannong Cao, *Hong Kong Polytechnic University, Hong Kong, Hong Kong*
Geoffrey Coulson, *Lancaster University, Lancaster, UK*
Falko Dressler, *University of Erlangen, Erlangen, Germany*
Domenico Ferrari, *Università Cattolica Piacenza, Piacenza, Italy*
Mario Gerla, *UCLA, Los Angeles, USA*
Hisashi Kobayashi, *Princeton University, Princeton, USA*
Sergio Palazzo, *University of Catania, Catania, Italy*
Sartaj Sahni, *University of Florida, Gainesville, USA*
Xuemin Shen , *University of Waterloo, Waterloo, Canada*
Mircea Stan, *University of Virginia, Charlottesville, USA*
Xiaohua Jia, *City University of Hong Kong, Kowloon, Hong Kong*
Albert Y. Zomaya, *University of Sydney, Sydney, Australia*

The LNICST series publishes ICST's conferences, symposia and workshops.

LNICST reports state-of-the-art results in areas related to the scope of the Institute.

The type of material published includes

- Proceedings (published in time for the respective event)
- Other edited monographs (such as project reports or invited volumes)

LNICST topics span the following areas:

- General Computer Science
- E-Economy
- E-Medicine
- Knowledge Management
- Multimedia
- Operations, Management and Policy
- Social Informatics
- Systems

Telex M. N. Ngatched · Isaac Woungang ·
Jules-Raymond Tapamo · Serestina Viriri
Editors

Pan-African Artificial Intelligence and Smart Systems

Third Pan-African Conference, PAAISS 2024
Durban, South Africa, December 4–6, 2024
Proceedings, Part II

Editors

Telex M. N. Ngatched 
McMaster University
Hamilton, ON, Canada

Isaac Woungang 
Toronto Metropolitan University
Toronto, ON, Canada

Jules-Raymond Tapamo 
University of KwaZulu-Natal
Durban, South Africa

Serestina Viriri 
University of KwaZulu-Natal
Durban, South Africa

ISSN 1867-8211

ISSN 1867-822X (electronic)

Lecture Notes of the Institute for Computer Sciences, Social Informatics
and Telecommunications Engineering

ISBN 978-3-031-94438-3

ISBN 978-3-031-94439-0 (eBook)

<https://doi.org/10.1007/978-3-031-94439-0>

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2025

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Preface

We are delighted to introduce the proceedings of the third edition of the Pan-African Artificial Intelligence and Smart Systems Conference (PAAISS 2024). This conference brought together leading academic research scientists, industry practitioners, independent scholars, and innovators from across the African continent and the world, to explore, exchange, and discuss the challenges and opportunities of harnessing the capabilities of Artificial Intelligence (AI) and Smart Systems (SS), which have emerged as the engine of the next wave of future innovations.

The theme of PAAISS 2024 was “Advancing AI research in Africa”. The technical program of PAAISS 2024 consisted of 39 full papers, carefully selected from 103 submissions, for oral presentation sessions at the conference tracks. All the submissions were full papers with no invited papers. The peer review process was conducted with the highest standards of academic rigor and integrity. All submissions underwent a rigorous single-blind peer review process, where each submission was reviewed by a minimum of 3 reviewers, who assessed the contributions based on their originality, scientific merit, relevance to the conference themes, and overall quality. In keeping with our commitment to fairness and transparency, any contributions co-authored by members of the conference committee were handled with particular care. To avoid any potential conflicts of interest, these submissions were reviewed by external reviewers who had no affiliation with the authors or the committee member. This ensured an objective evaluation of all contributions, maintaining the integrity of the peer review process.

Apart from the presentations of high-quality technical papers, the technical program also featured a welcoming address by Neil Koorbanally, acting DVC of Research, University of KwaZulu-Natal, South Africa, an address by Thomas Ndousse-Fetter, Imhotep Geostrategic Innovations, USA, and five keynote speeches. First, we would like to extend our heartfelt thanks to Neil Koorbanally and Thomas Ndousse-Fetter, whose words set the stage for a thought-provoking and inspiring conference, capturing the essence of the themes we explored throughout the event, and challenging us to think critically about the future of AI and SS in Africa. Then, we would like to extend our deepest thanks to our distinguished keynote speakers, whose insightful and stimulating presentations enlightened the conference. Their expertise and vision sparked rich discussions. Their contributions truly were a highlight of this event, and we are incredibly grateful for their participation. The five keynote presenters were Michael Felsberg, Linköping University, Sweden; Bigomokero Antoine Bagula, University of Western Cape, South Africa; Sudeep Tanwar, Nirma University, India; Foutse Khomh, Polytechnique Montréal, Canada, and Jerry John Kponyo, Kwame Nkrumah University of Science and Technology, Ghana.

An international conference of this size requires the support and hard work of many people to produce a successful PAAISS 2024 technical program and conference proceedings. Our Steering Committee chair, Dr. Thomas Ndousse-Fetter was essential for the success of the conference. We sincerely appreciate his vision, constant support, and

guidance. Our sincere appreciation also goes to the excellent Organizing Committee, whose tireless work behind the scenes ensured the success of this conference. From the early stages of planning to the final execution, their commitment to detail, organization, and the overall success of the event has been essential. We are grateful for their vision and dedication in making this conference a reality.

We are equally thankful to the various Track Chairs and Co-chairs, who played a crucial role in curating the content and overseeing the sessions. Their expertise and leadership helped shape the high quality of the program and facilitated engaging discussions across the various themes of the conference.

Our heartfelt appreciation also goes to the Technical Program Committee members, whose meticulous, rigorous, and timely review of the submitted papers ensured that only the highest-quality research was presented at the conference and made it into these proceedings. Their valuable feedback and constructive critiques have significantly elevated the caliber of this year's proceedings, and we are grateful for their time and effort. We also would like to express our sincere thanks to the authors of the papers featured in these proceedings. Your hard work, creativity, and dedication have made this conference a forum for sharing new ideas and advancing the frontiers of AI and SS. Without your contributions, this event would not have been possible, and we are grateful for the opportunity to showcase your research.

Finally, we would like to express our sincere gratitude to our hosts and sponsors, the University of KwaZulu-Natal and the Convention Bureau of Tourism and Film KwaZulu-Natal. Their generous support not only made the conference possible but also allowed us to enhance the overall experience for all participants. Special thanks to Jules-Raymond Tapamo, Serestina Viriri, and the University of KwaZulu-Natal administrators who led the local organization. We are also grateful to the Southern Sun Elangeni Maharani hotel for hosting the conference for in-person sessions.

We strongly believe that the PAAISS conference will continue in the coming years to be an excellent forum for all researchers, developers, and practitioners to discuss the related up-to-date science, technology, and practical aspects of Artificial Intelligence and Smart Systems. We also expect that the future PAAISS conferences will be as successful and stimulating, as shown by the contributions presented in this volume. We do hope that you will enjoy reading these proceedings.

Isaac Woungang
Telex M. N. Ngatched
Jules-Raymond Tapamo
Serestina Viriri
Thomas Ndousse-Fetter

Organization

Steering Committee

Thomas Ndousse-Fetter	Imhotep Research Institute, USA
Isaac Woungang	Toronto Metropolitan University, Canada
Telex M. N. Ngatched	McMaster University, Canada
Jules-Raymond Tapamo	University of KwaZulu-Natal, South Africa
Serestina Viriri	University of KwaZulu-Natal, South Africa

Organizing Committee

General Chair

Thomas Ndousse-Fetter	Imhotep Geostrategic Innovations, USA
-----------------------	---------------------------------------

General Co-chairs

Isaac Woungang	Toronto Metropolitan University, Canada
Serestina Viriri	University of KwaZulu-Natal, South Africa

TPC Chairs and Co-chairs

Telex M. N. Ngatched	McMaster University, Canada
Jules-Raymond Tapamo	University of KwaZulu-Natal, South Africa

Sponsorship and Exhibit Chair

Jules-Raymond Tapamo	University of KwaZulu-Natal, South Africa
----------------------	---

Local Chairs and Co-chairs

Tom Walingo	University of KwaZulu-Natal, South Africa
Jean Vincent Fonou Dombeu	University of KwaZulu-Natal, South Africa

Award Chair

Pius Adewale Owolawi	Tshwane University of Technology, South Africa
----------------------	--

Publicity and Social Media Chairs and Co-chairs

Justice Owusu Agyemang	Kwame Nkrumah University of Science and Technology, Ghana
Paul Python	University Cheikh Anta Diop of Dakar, Senegal

Publications Chair

Telex M. N. Ngatched	McMaster University, Canada
----------------------	-----------------------------

Web Chair

Justice Owusu Agyemang	Kwame Nkrumah University of Science and Technology, Ghana
------------------------	--

Panels Chairs and Co-chairs

Sylvester B. Aboagye	University of Guelph, Canada
Jules M. Moualeu	University of the Witwatersrand, South Africa

Track Co-chairs

Mandla Victor Gwetu	Stellenbosch University, South Africa
Jean Vincent Fonou Dombeu	University of KwaZulu-Natal, South Africa
Edgar Jembere	University of KwaZulu-Natal, South Africa
Pius Adewale Owolawi	Tshwane University of Technology, South Africa
Olutayo O. Oyerinde	University of the Witwatersrand, South Africa
Okuthe Paul Kogeda	University of KwaZulu-Natal, South Africa
Alain Richard Ndjiongue	University of the Witwatersrand, South Africa
Clement Nyieranda	University of the Western Cape, South Africa
Tom Walingo	University of KwaZulu-Natal, South Africa
Hongjun Xu	University of KwaZulu-Natal, South Africa
Jules M. Moualeu	University of the Witwatersrand, South Africa
Sylvester B. Aboagye	University of Guelph, Canada

Technical Program Committee

A. B. M. Bodrul Alam
 Faria Khandaker
 Anish Kurien
 Herve Frackin
 Nico Steyn
 Omar Maraqa
 Kingsley A. Ogudo
 Akintunde Ayodeji Alonge
 Sanjay Kumar Dhurandher
 Dane Brown
 Stones Chindipha
 Vinesh Kumar
 Bhekisizwe Mthethwa
 Yuval Genga
 Hamed Aly
 Megha Gupta
 Oluwaseyi Babalola

Clopas Kwenda
 Khulumani Sibanda
 Weston Govere
 Thambo Nyathi
 Mthulisi Velepini
 Dwane Brown
 Pravesh Ranchod
 Alok Shukla
 Sourabh Solanki
 Deemah Tashman
 Pedro Gorla Silva
 Collins Leke
 Majid H. A. Khoshafa
 Jackson Phiri
 Mehrdad Ghaziasgar
 Pakiso Khomokhoana
 Vitalis Ozianyi
 Patrick Nkwari
 Narushan Pillay
 Frank Igboamalu
 Boriane Tchaleu
 Carl Kugblenu

Algoma University, Brampton Campus, Canada
 Algoma University, Brampton Campus, Canada
 Tshwane University of Technology, South Africa
 University of Le Havre, France
 Tshwane University of Technology, South Africa
 McMaster University, Canada
 University of Johannesburg, South Africa
 University of Johannesburg, South Africa
 University of Delhi, India
 Rhodes University, South Africa
 Rhodes University, South Africa
 University of Delhi, India
 University of KwaZulu-Natal, South Africa
 University of the Witwatersrand, South Africa
 Dalhousie University, Canada
 University of Delhi, India
 Cape Peninsula University of Technology, South Africa
 Africa
 Great Zimbabwe University, Zimbabwe
 Walter Sisulu University, South Africa
 Arrupe Jesuit University, Zimbabwe
 University of Pretoria, South Africa
 University of Limpopo, South Africa
 Rhodes University, South Africa
 University of the Witwatersrand, South Africa
 University of the Witwatersrand, South Africa
 University of Luxembourg, Luxembourg
 Polytechnique Montréal, Canada
 Lappeenranta University of Technology, Finland
 University of Johannesburg, South Africa
 McMaster University, Canada
 University of Zambia, Zambia
 University of the Western Cape, South Africa
 University of the Free State, South Africa
 Strathmore University, Kenya
 University of Toronto, Canada
 University of KwaZulu-Natal, South Africa
 University of Johannesburg, South Africa
 University of Johannesburg, South Africa
 Aalto University, Finland

Mohamed Selim	Memorial University of Newfoundland, Canada
Saba Asaad	York University, Canada
Fanoro Mokesioluwa	University of Johannesburg, South Africa
Sandile Mhlanga	University of Johannesburg, South Africa
Roger Kwao Ahiadormey	Institute for Scientific and Technological Information, Council for Scientific and Industrial Research, Ghana
Elie Ngomseu Mambou	John Abbott College, Canada
Lawrence Githuari	University of Nairobi, Kenya
Khulumani Sibanda	University of Fort Hare, South Africa
Hossana Twinomurizi	University of Johannesburg, South Africa
Oludayo O. Olugbara	Durban University of Technology, South Africa
Dustin Van der Haar	University of Johannesburg, South Africa
Cheikh Ahmadou Bamba Gueye	University Cheikh Anta Diop of Dakar, Senegal
Assane Gueye	Carnegie Mellon University Africa, Rwanda
Jean Marie Demebele	Gaston Berger University of Saint-Louis, Senegal
Youssou Faye	University Assane Seck of Ziguinchor, Senegal
Cheikh Sarr	University Iba Der Thiam of Thies, Senegal
Mamadou Bousso	University Iba Der Thiam of Thies, Senegal
Youssou Dieng	University Assane Seck of Ziguinchor, Senegal
Cheikh Talibouya Diop	Gaston Berger University of Saint-Louis, Senegal
Moussa Lo	Université virtuelle du Sénégal, Senegal
Dame Diongue	Gaston Berger University of Saint-Louis, Senegal
Ousmane Sall	Université virtuelle du Sénégal, Senegal
Ibrahima Niang	University Cheikh Anta Diop of Dakar, Senegal
Gervais Mendy	Ecole supérieure Polytechnique de Dakar, Senegal
Ado Adamou Abba Ari	University of Maroua, Cameroon
Hippolyte Michel T. Kenfack	University of Yaoundé I, Cameroon
Moussa Diallo	École supérieure Polytechnique de Dakar, Senegal
Mourad Gueroui	Université de Versailles Saint-Quentin en Yvelines, France
Jupiter Romuald N. Bakakeu	Alteryx, USA and University of Erlangen, Germany
Raphael Angulu	Masinde Muliro University of Science and Technology, Kenya
Glaucio H. S. Carvalho	Brock University, Canada
Andrea Visconti	Milan University, Italy
Herve Frackin	University of Le Havre, France
Lawrence Githuari	University of Nairobi, Kenya
Ismail Ateya	University of Strathmore, Kenya
Shibwabo Benard	University of Strathmore, Kenya
Mohammed-Sani Abdulai	Advanced Information Technology Institute, Ghana

Shadrack Maina Mambo	Kenyatta University, Kenya
Muhammad Fazal Ijaz	Sejong University, South Korea
Ayodele Adebisi	Covenant University, Nigeria
Muhammad Haroon Yousaf	University of Engineering & Technology Taxila, Pakistan
David Celestin Faye	Gaston Berger University of Saint-Louis, Senegal
Seydina Moussa Ndiaye	Université Virtuelle du Senegal, Senegal
Mary Ahuna	Technical University of Kenya, Kenya
Franklin Tchakounté	University of Ngaoundere, Cameroon
Mehrdad Tirandazian	Toronto Metropolitan University, Canada
Ignace Tchangou Toudjeu	Tshwane University of Technology, South Africa
Jimmi Talla Mbé	University of Dschang, Cameroon
Jules Waku	University of Yaoundé 1, Cameroon
Absalom Ezugwu	University of KwaZulu-Natal, South Africa
Justine Daramola	Cape Peninsula University of Technology, South Africa

Contents – Part II

Artificial Intelligence Theory and Methods

Scalable XAI: Towards Explainable Machine Learning Models in Distributed Systems	3
<i>Ibomoiye Domor Mienye, Theo G. Swart, and George Obaido</i>	
An Ensemble of Yolov5 Models in Real-Time Object Detection in Low Resource Settings	17
<i>Nkalubo Lenard Byenkya, Nakibuule Rose, and Okila Nixon</i>	
Hybridizing Deep Neural Networks and Machine Learning Models for Natural Scene Image Classification	36
<i>Clopas Kwenda, Mandlenkosi Victor Gwetu, and Jean Vincent Fonou Dombeu</i>	
Ensemble CNNs for Solar Flare Image Classification	55
<i>Mangaliso Mngomezulu and Mandlenkosi Gwetu</i>	
Natural Language Processing in Automatic Grading of Assessments in Higher Education: A Systematic Literature Review	70
<i>Lizzy Ofusori, Tebogo Bokaba, and Siyabonga Mhlongo</i>	
An Attention-Based Deep Learning Model for Term Extraction from Text Using BERT	87
<i>Tsitsi Zengeya, Jean Vincent Fonou Dombeu, and Mandlenkosi Gwetu</i>	

Artificial Intelligence and Smart Systems

Temporal Analysis of Social Concerns on African Social Media: Insights from Topics, Themes, and Sentiments	105
<i>Harriet Sibitenda, Awa Diattara, Assitan Traore, Elke Rundensteiner, and Cheikh Ba</i>	
Real-Time Object Detection Using an Ensemble of One Stage and Two Stage Object Detection Models with Dynamic Fine-Tuning Using Kullback-Leibler Divergence	125
<i>Nkalubo Lenard Byenkya, Nakibuule Rose, and Okila Nixon</i>	

Hybrid Approach for Image Processing Based on Convolutional Neural Network in Facial Recognition System	144
<i>Ibrahim Aoudou Pouamoun, Justin Moskolai Ngossaha, René Ndoundam, Francis Yongwa Dtissibe, Abdelhak Mourad Gueroui, and Ado Adamou Abba Ari</i>	
Network Anomaly Detection System Using an Attention-Based GNN	164
<i>Amir Mohammadi Bagha, Isaac Woungang, Issa Traore, Danda B. Rawat, Sudeep Tanwar, and Azin Hassanalizadeh</i>	
Truckchain: A Blockchain-Powered IoT Real-Time Tracking System for Fuel Supply Chain Management	177
<i>Cansa Kayembe Amaury, Kabemba Ntumbwa Jonathan, and Mpyana Mwamba Merlec</i>	
Food Safety 4.0: The Future of Food Safety Leveraging Industry 4.0 Technologies	194
<i>Eustace M. Dogo, Tebogo Bokaba, Hussaini A. Makun, Ahmed Aliyu, and Ponjul Bali Kparbong</i>	
Soil Drainage Classification Using Machine Learning Models: A Comparative Study	213
<i>Mouhamadou Lamine Diakhame, Diarra Bousso Dieng, Joshua Nougaret, and Jonathan Mwaura</i>	
A Comparative Analysis of FA1→3 and Lévy-Flight Based FA1→3 with Applications in Credit Card Fraud Using SMOTE Data Augmentation	227
<i>Trevor Vambe, Mandlenkosi Gwetu, and Serestina Viriri</i>	
Juro: A Retrieval-Augmented Generation AI Chatbot for Enhancing Legal Information Access in Resource-Constrained Settings	260
<i>Bernard Ngandu, Landry Mbale, and Antoine Bagula</i>	
Campus WiFi Demand Prediction: A Case Study on KNUST Campus	275
<i>Justice Owusu Agyemang, Manhyia Gideon Aboagye, Awumee Gabriel Selorm, Somuah Obed, Addai Clinton, and Henyo Chester Elikplim</i>	
Remote Sensing and Artificial Intelligence	
A Particle Swarm Optimization-Long-Short Term Memory (PSO-LSTM) Hybrid Model for Forecasting Global Horizontal Solar Radiation	291
<i>Blessing Olatunde Abisoye, Yanxia Sun, and Wang Zenghui</i>	

Modelling Road Networks with Node Structural Features and Graph Convolutional Networks	309
<i>Mohale Molefe</i>	
Leveraging Machine Learning and Climate Data for Enhanced Annual Crop Production Forecasts in Senegal	326
<i>Kpangni Alex Jérémie Koua, Fatou Cire Emilie Sané, Michael Warren, and Jonathan Mwaura</i>	
Road Obstacle Detection Using YOLO Algorithm Based on Attention Mechanism	347
<i>Bafokeng Lekola and Serestina Lekola</i>	
Machine Learning-Based System for Automated Crop Damage Detection and Classification in Insurance Claims	369
<i>Abdou Karim Kandji, Kpêatchéhoué Merveille Santi Zinsou, Thomas Henahan, and Jonathan Mwaura</i>	
Author Index	387

Contents – Part I

Artificial Intelligence in Medicine

A Real-Time Monitoring System for Neurodegenerative Disease Progression Utilizing Generative Adversarial Networks	3
<i>Boriane Y. Tchaleu, Alain R. Ndjiongue, and Collins A. Leke</i>	
Detection and Classification of Prostate Cancer Using Lightweight MobileNet	15
<i>Olushola Olawuyi</i>	
A Comparative Analysis of Performance Metrics in Deep Learning Models for Brain Tumour Detection Using MRI Scans	34
<i>Larisha Moodley, Yuvika Singh, and Okuthe P. Kogeda</i>	
Prostate Cancer Detection and Classification Using Deep Learning Pretrained Model	54
<i>Olushola Olawuyi</i>	
Empowering Healthcare Queries: A Chatbot for Converting Text to SQL	72
<i>Nyashadzashe Tamuka and Khulumani Sibanda</i>	
HEALeB: Healthcare 4.0 Enabled by IoT, AI, and Blockchain	83
<i>Henriette Kyangwa, Landry Mbale, and Antoine Bagula</i>	

Smart Systems Enabling Technologies

Comparative Analysis of AI-Based Algorithms for the Detection of Tuberculosis Bacilli on Digitised Human Tissue	97
<i>Kapongo D. Lumamba, Threnesan Naidoo, Mpumelelo Msimnag, Gordon Wells, Adrie J. C. Steyn, and Mandlenkosi V. Gwetu</i>	
Classical vs Deep Learning Models: A Study on Maize Leaf Diseases Classification	121
<i>Chimango Nyasulu, Cecilia Tivir, and Jonathan Mwaura</i>	
Implementing a Novel Cybersecurity Algorithm for Smart Grid SCADA System	133
<i>Kholofelo H. Halefose, Topside E. Mathonsi, Tonderai Muchenje, and Deon Du Plessis</i>	

Improving Face Recognition from a Single Image 152
Justice Owusu Agyemang, Desmond Delali Atakpla, Awumee Gabriel Selorm, Thomas Boadu, Martin Kodua-Basoa, and Christabel Bemah Darko

Development of Automated Banana Sorter Using Computer Vision Techniques 166
Semakaleng Seretloa, Pius Owolawi, and Chunling Tu

Evaluating the Effectiveness of Current Countermeasures Against Man-In-The-Middle Attack on Domain Name System 177
Phandavhudzi Rotshidzwa, Moila Ramahlapane Lerato, and Velempini Mthulisi

Artificial Intelligence-Enabled Communication Systems

Predicting Coronal Hole Activity: Key to Mitigating Space Weather Impacts ... 193
Tarek Alsolame, B. D. S. Aritra, Enock Niyonkuru, Stephen Antogiovanni, and Chandranil Chakrabortii

Building Mapping for Implementing an Object-Oriented Model of Educational Infrastructure Usage at Jean Lorougnon Guédé (UJLoG) University in Central-Western Côte d’Ivoire 210
N’guessan Patrice Akoguhi, Kpangni Alex Jérémie Koua, Egomli Stanislas Assohoun, N’dri Pascal Kouame, and Koffi Fernand Kouame

Optical RIS in Indoor VLC Environments: What are the Suitable Materials 228
Frank Igboamalu, Alain Ndjiongue, Khmaies Ouahada, and Ling Cheng

Single- vs Multi-connectivity in Movable Access Point-Enabled Visible Light Communication Systems 240
Zohaib Saghir, Raphael Ahiaklo-Kuz, and Sylvester Aboagye

A Review on VLC-Based Indoor Positioning Technologies 252
Felipe Bueno

Evolving Trends in the Application of Artificial Intelligence to Support the Sustainable Development Goals: A Bibliometric Analysis 275
Kudakwashe Maguraushe, Patrick Ndayizigamiye, and Tebogo Bokaba

Author Index 293

Artificial Intelligence Theory and Methods



Scalable XAI: Towards Explainable Machine Learning Models in Distributed Systems

Ibomoiye Domor Mienye^{1(✉)}, Theo G. Swart¹, and George Obaido²

¹ Institute for Intelligent Systems, University of Johannesburg, Auckland Park, Johannesburg 2006, South Africa
{ibomoiyem,tgswart}@uj.ac.za

² Center for Human-Compatible Artificial Intelligence (CHAI), Berkeley Institute for Data Science (BIDS), University of California, Berkeley, CA 94720, USA
gobaido@berkeley.edu

Abstract. The application of artificial intelligence (AI) has grown significantly due to recent advances in machine learning (ML). However, deploying ML models within distributed computing environments introduces substantial challenges, particularly in terms of scalability and explainability. This review addresses the concept of scalable explainable AI (XAI), focusing on effective methodologies for deploying XAI across various distributed frameworks. The study aims to advance the development of transparent, reliable, and scalable AI applications, thereby ensuring greater trust and broader adoption in diverse operational environments.

Keywords: Artificial intelligence · distributed systems · machine learning · XAI

1 Introduction

Explainable artificial intelligence (XAI) presents a crucial transformation in how we understand, trust, and manage AI systems [1, 2]. The transparency and interpretability of AI are now technical requirements and are increasingly seen as indispensable due to ethical and regulatory considerations [3]. Such transparency ensures that AI decisions can be understood and trusted by users and aligns with broader goals of accountability in automated systems. The significance of XAI thus spans various sectors, from healthcare diagnostics to financial services and smart grids, where the decision-making processes of AI systems must be clear and justifiable [4].

Meanwhile, as data volumes grow and computational models become more complex, the ability to scale these systems becomes crucial [5]. Scalability impacts the efficiency and speed of data processing and the overall effectiveness of ML solutions. It ensures that as datasets grow and models become more intricate, the performance and accuracy of these systems do not diminish but instead

improve or maintain their effectiveness. This led to the emergence of distributed computing, a vital technology in modern AI. It enables the processing and analysis of large-scale data across clusters of computers [6, 7]. Frameworks such as Apache Hadoop and Apache Spark have become synonymous with scalable data processing, which plays a crucial role in the deployment and operational efficiency of large-scale ML models. These technologies facilitate the management of big data and ensure that AI applications remain robust and scalable across various computing environments.

As AI models become more integrated into critical aspects of society, including healthcare, finance, and public safety, the demand for transparency and accountability in these systems has increased. Regulatory bodies worldwide are beginning to impose stricter guidelines that mandate explainability in AI systems to ensure they are fair, transparent, and accountable. For instance, the European Union’s General Data Protection Regulation (GDPR) includes provisions that can be interpreted to require explainable AI processes in certain contexts [8].

Furthermore, AI technologies are evolving rapidly, with advancements in ML models that are increasingly complex and harder to interpret [9]. This complexity challenges traditional explanation methods and demands innovative approaches to make these systems understandable without sacrificing performance. Scalable XAI is critical in this context because it seeks to bridge the gap between advanced ML techniques and the practical need for clear, understandable AI decision-making processes across large-scale, distributed environments.

This review aims to synthesize current research on scalable XAI, examining how explainable AI models are deployed across different distributed computing frameworks. It aims to address the inherent challenges and present potential solutions to deploying and interpreting XAI at scale. Furthermore, the review will explore future research directions and innovations needed to advance the field, making substantial contributions to academic research and practical applications in scalable XAI. In addition, the rapid growth of big data and the increasing adoption of cloud and distributed computing technologies have amplified the need for scalable XAI solutions that can operate efficiently across different frameworks and infrastructures. This review is, therefore, crucial to advancing the understanding of how XAI can be effectively implemented at scale, addressing both technical challenges and the broader implications for society.

The rest of the paper is organised as follows: Sect. 2 discusses the theoretical frameworks of explainable AI, including common techniques, and evaluation metrics. Section 3 presents an overview of distributed computing frameworks, while Sect. 4 discusses challenges in scalable XAI deployment. Section 5 presents methodologies for scaling XAI, Sect. 6 discusses future research directions, and Sect. 7 concludes the paper.

2 Theoretical Framework of XAI

XAI aims to make the outcomes of AI models transparent, understandable, and trustworthy. The core principles of XAI are centred around three fundamental

concepts: transparency, intelligibility, and accountability. Transparency ensures that the operations and results of AI models are visible and accessible to a wide range of stakeholders, including developers, end-users, and regulators, without requiring specialised knowledge to interpret these processes [3]. This openness is crucial for fostering trust and facilitating the broader adoption of AI technologies.

On the other hand, Intelligibility relates to the ability of AI systems to express their functionalities and rationales in human-comprehensible terms. This means that the system should be able to explain or present its behaviour and the logic behind its decisions in an understandable way to its users [10]. This aspect is essential in complex systems where decisions need to be demystified for non-expert users. Lastly, accountability in XAI refers to the clear attribution of decisions made by AI systems [3]. This principle ensures that AI operations are transparent and understandable and that there is a clear line of responsibility for the actions taken by these systems. This is critical for ethical considerations, where it is necessary to pinpoint where the responsibilities lie, particularly in scenarios where decisions may significantly impact individuals or communities.

2.1 Common Techniques in XAI

Different techniques and tools have been developed to aid the implementation of explainability in AI systems. These tools and methods vary widely in their approach and the type of explanations they generate:

Feature Importance Methods. Techniques such as SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) play a crucial role in XAI by quantifying the contribution of each feature to the decision made by the model. SHAP leverages concepts from cooperative game theory, specifically the Shapley value, to assign an “importance” value to each feature for a particular prediction [11]. The Shapley value ϕ_i for a feature i is defined as:

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (v(S \cup \{i\}) - v(S))$$

where N is the set of all features, S is a subset of features excluding i , and $v(S)$ is the value function that measures the prediction’s accuracy with subset S of features. This formulation ensures a fair distribution of contributions based on the marginal contribution of each feature across all possible combinations.

Conversely, LIME approximates the local behaviour of complex models with simpler models that are easier to understand. It modifies the input data by perturbing the feature values and observes the changes in the output [12]. The explanation model, typically a linear model, is then trained on this new dataset to approximate the predictions of the original model locally. The coefficients of the linear model provide an interpretable representation of feature importance:

$$\hat{y} = \beta_0 + \sum \beta_i x_i$$

where $\hat{y}$ is the prediction, β_i are the coefficients indicating the importance of each feature x_i , and β_0 is the intercept. This model is used to explain individual predictions by showing which features contribute most significantly to the model’s output, and in what way. Both SHAP and LIME provide powerful tools for interpreting the decisions of machine learning models.

Model Visualization. This technique involves tools that visualise the inner workings and decision paths within models, which is especially useful in complex models like neural networks. Visualisation can include displaying layers, weights, neurons, and activations, allowing users to visually track how inputs are transformed into outputs [13]. This method is crucial for diagnosing and debugging model behaviour, particularly deep learning.

Surrogate Models. Surrogate models are simplified approximations that mimic the behaviour of more complex models. These models are particularly beneficial when the original model is too complex to be directly interpretable, allowing for an easier-to-understand model to stand in as an explanatory proxy [14]. Mathematically, a surrogate model is typically trained to approximate the decision function of the original model by using a simpler or more interpretable model structure, such as linear regression or decision trees. For a given complex model with a decision function $f(x)$, where x is an input feature vector, the surrogate model $g(x)$ is trained on a dataset generated by sampling from the distribution of x and obtaining corresponding outputs from $f(x)$. The objective is to minimise the discrepancy between the predictions of the complex model and the surrogate model, typically measured by a loss function L . The formulation can be represented as:

$$g(x) = \arg \min_g L(f(x), g(x))$$

where L is a suitable loss function, such as mean squared error for regression tasks or cross-entropy loss for classification tasks. The choice of the surrogate model $g(x)$ and the loss function L depends on the specific requirements for interpretability and the nature of the original model’s outputs.

2.2 Evaluation Metrics for Explainability

Evaluating the effectiveness of XAI systems involves assessing how well the explanations meet users’ needs for understanding and trusting AI systems. To measure the quality and utility of explanations, several key metrics are commonly used:

Fidelity. Fidelity measures how accurately explanations reflect the behaviour of the AI models. High fidelity in an explanation means that it truly reflects the operations and outcomes of the model it explains, ensuring that the user’s understanding is based on an accurate representation of how the model functions [15]. Mathematically, it can be expressed as:

$$\text{Fidelity} = 1 - \frac{1}{N} \sum_{i=1}^N |f(x_i) - g(x_i)|$$

where $f(x)$ is the original model’s output, $g(x)$ is the output of the surrogate model used for explanation, and N is the number of samples. A lower value indicates a higher fidelity, meaning the explanation closely mirrors the model’s behaviour.

Consistency. Consistency assesses whether similar explanations are generated for similar instances or across different models performing similar tasks [15]:

$$\text{Consistency} = \frac{1}{N} \sum_{i=1}^N \mathbf{1}(e(x_i) = e(x'_i))$$

where $e(x)$ represents the explanation for input x , x'_i is a perturbed version of x_i such that $f(x_i) = f(x'_i)$, and $\mathbf{1}$ is the indicator function. High consistency means that the model provides stable explanations across similar scenarios.

Comprehensibility: This metric evaluates how easily human users can understand the explanations [16]. No matter how accurate, an explanation must be interpretable to the end user to be considered beneficial. Comprehensibility involves using clear, understandable language or visualisations that make sense to users without specialised knowledge, thereby democratising the use of AI technologies.

Coverage: Coverage refers to the extent to which the explanations apply to various situations the AI system may encounter. This metric looks at the robustness of the explanation across all possible inputs and scenarios in which the model operates [17]. High coverage ensures that explanations are not just applicable to a narrow set of cases but are broadly applicable, enhancing the overall reliability of the AI system.

3 Distributed Computing Frameworks

Distributed computing plays a crucial role in modern AI by enabling the processing and analysis of large-scale data across networked computers and servers. Various distributed computing frameworks have become essential in the development and deployment of AI applications due to their ability to handle vast amounts of data and complex computations, including:

3.1 Apache Hadoop

Originally designed for web-scale indexing, Hadoop has evolved into a robust framework that supports the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale from a single server to thousands of machines, each offering local computation and storage. The framework's core includes a storage part, known as Hadoop Distributed File System (HDFS), and a processing part called MapReduce [18]. Hadoop is widely used for batch processing of data, making it suitable for tasks where data size and complexity require substantial computational power and storage.

3.2 Apache Spark

Developed to address the limitations of MapReduce in processing time-sensitive data, Spark provides an in-memory data processing capability that significantly speeds up the analytical process [19]. Known for its speed and API ease, Spark is a fast and general-purpose cluster computing system. It supports high-level APIs in Java, Scala, Python, and R, along with an optimized engine that supports general computation graphs for data analysis. Spark is particularly beneficial for real-time analytics and ML tasks requiring quick iterative data access.

3.3 Google's TensorFlow with TensorFlow Extended

As a more specialized framework, TensorFlow offers comprehensive, flexible ecosystems of tools, libraries, and community resources that let researchers push the state-of-the-art in ML, and developers easily build and deploy ML-powered applications [20]. TensorFlow Extended (TFX) is an end-to-end platform designed to deploy production ML pipelines - from data ingestion and preparation to training, validation, and serving. It is particularly effective for deploying deep learning models in distributed environments, optimizing computational efficiency and allowing for scalable, robust deployment of complex models.

4 Challenges in Scalable XAI Deployment

The deployment of scalable XAI systems presents several challenges that stem from data management complexities, model intricacies, and infrastructure requirements. These challenges must be addressed to ensure effective scaling of XAI across different platforms and environments.

4.1 Data Management Issues

Effective data management is critical for the successful deployment of AI systems, especially when scalability and explainability are required. Key issues include:

Volume, Velocity, and Variety. The three Vs of big data pose significant challenges in XAI. High volume requires scalable storage solutions that efficiently accommodate terabytes or even petabytes of data [21]. High velocity demands robust real-time processing capabilities to handle streaming data that must be analyzed as it is captured. High variety involves managing and integrating diverse data types and sources, from structured numerical data to unstructured text, images, or videos. Each aspect complicates the training and explanation processes within AI systems, as they require sophisticated algorithms to ensure accurate and interpretable results.

Data Privacy and Security Concerns. As AI systems handle increasingly sensitive information, ensuring data privacy and security becomes paramount. This involves implementing robust encryption methods to secure data at rest and in transit [22, 23]. Secure data access protocols must be enforced, along with comprehensive auditing mechanisms to track who accesses the data and for what purpose. Compliance with international data protection regulations (e.g., GDPR in Europe and HIPAA in the United States) is crucial and often requires extensive documentation and regular audits to ensure practices meet legal standards. These requirements add layers of complexity to data handling procedures in XAI systems, impacting scalability as each layer of security may introduce additional processing overhead or restrictions on how data can be used.

4.2 Model Complexity and Management

As AI models become more complex, managing them effectively at scale and maintaining their interpretability pose additional challenges:

Scaling Complex AI Models. Modern AI, particularly deep learning, involves models that often contain millions of parameters and are built with complex, layered structures that can be inherently difficult to manage and scale [4]. These models require vast amounts of data and computational power for training, which can be challenging to distribute across multiple computing nodes without degradation in performance or accuracy. Techniques such as model parallelism, where different parts of a neural network are processed on different nodes, or data parallelism, where the same model is used across different nodes but with different data subsets, are commonly employed [24]. However, both approaches (shown in Fig. 1) require sophisticated coordination and high-bandwidth communication between nodes, adding to the complexity of the system’s infrastructure.

Maintaining Model Interpretability at Scale. As AI models become larger and more complex, they often become less transparent, turning into “black boxes” where it is unclear how inputs are transformed into outputs [2, 25]. This lack of transparency can be a significant barrier in applications where understanding the decision-making process is crucial. To counter this, techniques such

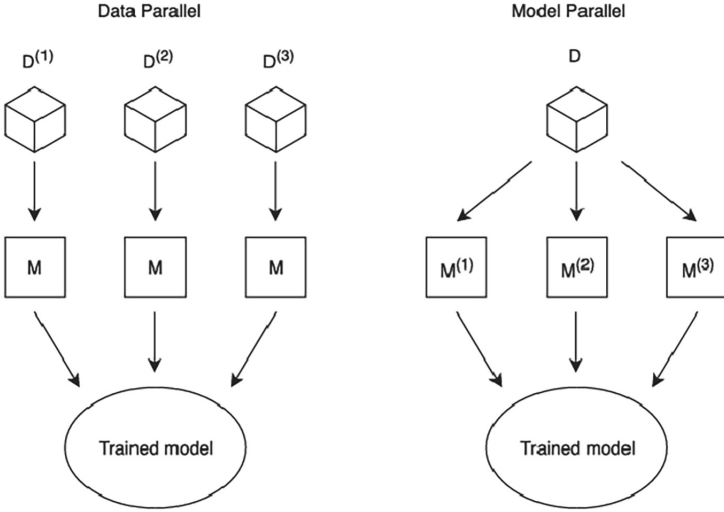


Fig. 1. Parallelism in distributed ML frameworks [4]

as simplifying model architectures, employing explainability-focused layers or methods within the models, or using post-hoc interpretation techniques like LIME or SHAP are utilized.

Trade-Offs Between Performance and Interpretability. Often, there is an inherent trade-off between the performance of an AI model and its interpretability. Highly accurate models like deep neural networks are typically less interpretable, whereas simpler models such as decision trees offer more transparency but may not achieve the same level of accuracy [26]. Balancing these aspects requires careful consideration of the model’s intended use and the criticality of explainability in its application, necessitating a tailored approach to model development and deployment in scalable environments.

4.3 Infrastructure and Resource Constraints

The infrastructure needed to support scalable XAI must be robust and flexible, yet it is often constrained by available resources, impacting the ability to deploy and scale AI systems effectively:

Computational Requirements. Scalable XAI systems demand significant computational power to process large datasets, train complex models, and generate explanations. This requirement can be a major limiting factor, particularly in resource-constrained environments such as small organizations or developing countries. Optimizing computational efficiency involves enhancing algorithm efficiency, adopting more powerful processing hardware, or utilizing cloud-based

solutions to handle peak loads without needing constant physical infrastructure upgrades [27, 28]. However, each of these solutions comes with trade-offs in terms of cost, latency, and dependency on external services, which can complicate the deployment and operational aspects of XAI.

Hardware and Software Dependencies. XAI systems often rely on specific types of hardware, such as GPUs, for processing deep learning algorithms, which can limit scalability if such resources are not readily available. Additionally, software dependencies, including the need for specific operating systems or proprietary software, can further restrict the deployment of XAI systems across different environments [2]. Ensuring that XAI frameworks are compatible with a wide range of technologies and that they can leverage the latest advancements in hardware is crucial. This includes adopting hardware-agnostic software architectures and considering open standards for software and data interoperability. Planning for hardware upgrades and anticipating the integration challenges associated with new technologies is essential to ensure seamless scalability.

5 Methodologies for Scaling XAI

Scalable XAI systems require robust methodologies to handle the increased complexity and data volumes typical of large-scale deployments. These methodologies span architectural approaches, algorithmic and software strategies, and framework-specific implementations.

5.1 Architectural Approaches

Effective architectural strategies are crucial for supporting scalable XAI systems. These strategies determine how data and computations are managed across different environments, directly impacting performance, scalability, and the feasibility of real-time explainability.

Centralized vs. Decentralized Architectures: Centralized architectures operate by managing all data and processing through a single central system. This model simplifies data governance and model management by centralizing the control and oversight of operations, making it easier to implement consistent policies and practices [29]. However, centralized systems can suffer from significant bottlenecks, where the central node becomes a point of failure or congestion, limiting scalability. They are typically suited for environments where data integration and security are more controlled and can be tightly managed.

Meanwhile, decentralized architectures distribute data and processing tasks across multiple nodes or locations. This structure enhances system scalability by spreading the computational load and storage demands, significantly improving fault tolerance and system resilience [29]. Decentralized systems require sophisticated coordination mechanisms and advanced data consistency strategies to

ensure that all nodes in the network synchronize correctly and that data integrity is maintained across different points. This architecture is preferable in scenarios requiring high availability and robustness and where data is generated in dispersed locations, such as large-scale sensor networks or multinational organizations.

Federated Learning Models. Federated learning represents a shift from the traditional centralized ML approaches to a more distributed methodology. The training data remains on local devices or servers (nodes) in this model, and only the model updates are transmitted to a central server [30]. The central server then aggregates these updates to improve the overall model, which is subsequently shared with the nodes. This approach is highly beneficial for XAI applications in privacy-sensitive areas, such as personal data analysis and healthcare, as it allows for generating explanations and model adjustments without exposing individual data points [31]. Federated learning reduces the need for large-scale data transfers, thereby decreasing the bandwidth requirements, which is essential for geographically distributed or bandwidth-constrained environments. Moreover, it supports more personalized model tuning and explanation generation at the node level, which can increase the relevance and accuracy of the explanations provided to end-users.

5.2 Algorithmic and Software Solutions

Optimizing algorithms and software is essential for efficiently operating scalable XAI systems. This involves enhancing the computational efficiency of the algorithms and ensuring that the software architecture supports scalability and maintainability. Model compression and simplification techniques such as pruning, quantization, and knowledge distillation are vital. Pruning involves removing unnecessary or redundant weights from a neural network, reducing the model size and computation required during inference, which can lead to improvements in speed and resource use [32]. This also helps enhance model interpretability by focusing on the most crucial connections. Quantization, which reduces the precision of the model’s parameters from floating-point numbers to integers, can significantly decrease model size and enhance inference speed, making it especially beneficial for deployment on edge devices with limited computational resources [33]. Knowledge distillation involves training a smaller “student” model to emulate a more extensive “teacher” model, simplifying the model while retaining much of its accuracy, thus aiding both performance and interpretability.

5.3 Framework-Specific Implementation Strategies

Tailoring XAI implementations to specific frameworks can maximize performance and compatibility, which is essential for ensuring that XAI systems scale efficiently across diverse environments. Customizing XAI techniques to different distributed frameworks involves several key strategies. Adapting to the unique

capabilities and limitations of each framework, such as Apache Spark, Hadoop, or cloud-based services like AWS or Google Cloud, is crucial. For instance, leveraging Spark’s in-memory data processing capabilities can significantly enhance the performance of explainability features in big data scenarios [34,35]. Additionally, optimizing data formats to suit the specific strengths of a framework, like using columnar storage formats such as Parquet for analytics on Hadoop and Spark, can improve data ingestion and processing speeds. Tuning the performance settings of each framework according to the specific demands of XAI tasks, like model training and real-time explanation generation, also plays a vital role in resource management and operational efficiency.

6 Future Research Directions

As explainable artificial intelligence continues to evolve, several key areas are critical for future research. These directions aim to address current limitations and expand the capabilities of XAI systems in scalable and distributed environments. Firstly, the development of more sophisticated explanation methods is crucial. Research should focus on enhancing context-aware explanation techniques, integrating environmental and situational data to ensure that explanations are relevant and specifically tailored to the circumstances surrounding AI decisions [36]. Additionally, creating adaptive explanation systems that can dynamically adjust based on user feedback will enhance the user-centric effectiveness of these systems. These systems would modify the detail or type of explanations based on user interactions to improve clarity and understanding.

Another vital area of research involves handling the dynamic nature of AI models, particularly those in production environments that continue to learn and adapt over time. Developing methods that can transparently track and explain these changes is essential for maintaining trust and accountability. Additionally, integrating causal inference methods into XAI represents a promising avenue for enhancing the depth and quality of explanations provided by AI models [17]. Unlike traditional explanation techniques that often focus on correlation, causal inference seeks to determine why certain decisions or outcomes occur due to specific inputs or actions. This approach could revolutionise XAI by moving beyond merely describing patterns observed in the data to elucidating the underlying causal mechanisms that drive AI decisions. This is crucial for applications where understanding the cause-and-effect relationship is as important as the outcome itself. For instance, in medical diagnostics, finance, or autonomous driving, knowing the causal factors behind AI decisions can provide critical insights that improve decision-making, enhance trust, and facilitate greater accountability.

Furthermore, there is a need for solutions that can deploy explainable AI at scale with minimal human oversight, which necessitates significant advancements in how explainability is integrated from the very outset of the AI development lifecycle. Focusing research on creating algorithms and architectures that embed explainability into the fabric of AI systems could transform how these technologies are perceived and utilised. Ensuring that every model generated is inherently explainable and designed to scale efficiently allows us to facilitate broader

deployment across diverse environments without compromising transparency or accountability.

7 Conclusion

This review has explored the various aspects of scalable explainable artificial intelligence, highlighting the critical role of advanced architectures, algorithms, and frameworks in facilitating the deployment of XAI across distributed computing environments. We have discussed the importance of developing XAI systems that are technically proficient, inherently interpretable, and trustworthy. The exploration of different architectural strategies, such as centralised and decentralised approaches, along with federated learning models, underscores the necessity to adapt and innovate continuously to meet diverse operational demands. Ultimately, the journey towards fully explainable and scalable AI systems is ongoing, and the insights gained from this review will contribute to shaping the future of XAI in a world increasingly driven by artificial intelligence.

References

1. Ahmed, I., Jeon, G., Piccialli, F.: From artificial intelligence to explainable artificial intelligence in industry 4.0: a survey on what, how, and where. *IEEE Trans. Industr. Inf.* **18**(8), 5031–5042 (2022)
2. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE Access* **6**, 52138–52160 (2018)
3. Felzmann, H., Fosch-Villaronga, E., Lutz, C., Tamò-Larrieux, A.: Towards transparency by design for artificial intelligence. *Sci. Eng. Ethics* **26**(6), 3333–3361 (2020)
4. Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., Rellermeyer, J.S.: A survey on distributed machine learning. *ACM Comput. Surv. (CSUR)* **53**(2), 1–33 (2020)
5. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: a review. *Neural Netw.* **113**, 54–71 (2019)
6. Schadt, E.E., Linderman, M.D., Sorenson, J., Lee, L., Nolan, G.P.: Computational solutions to large-scale data management and analysis. *Nat. Rev. Genet.* **11**(9), 647–657 (2010)
7. Gill, S.S., et al.: Ai for next generation computing: emerging trends and future directions. *Internet Things* **19**, 100514 (2022)
8. Hamon, R., Junklewitz, H., Malgieri, G., Hert, P.D., Beslay, L., Sanchez, I.: Impossible explanations? Beyond explainable AI in the GDPR from a covid-19 use case scenario. In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 549–559 (2021)
9. Lu, Y.: Artificial intelligence: a survey on evolution, models, applications and future trends. *J. Manag. Anal.* **6**(1), 1–29 (2019)
10. Sokol, K., Flach, P.: Explainability is in the mind of the beholder: establishing the foundations of explainable artificial intelligence. *arXiv preprint [arXiv:2112.14466](https://arxiv.org/abs/2112.14466)* (2021)

11. Lundberg, S.M., Lee, S.-I.: A unified approach to interpreting model predictions. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
12. Garreau, D., Luxburg, U.: Explaining the explainer: a first theoretical analysis of lime. In: *International Conference on Artificial Intelligence and Statistics*, pp. 1287–1296. PMLR (2020)
13. Alicioglu, G., Sun, B.: A survey of visual analytics for explainable artificial intelligence methods. *Comput. Graph.* **102**, 502–520 (2022)
14. Kumar, A., Patil, S., Kovacevic, A., Ponnusami, S.A.: Performance prediction and Bayesian optimization of screw compressors using gaussian process regression. *Eng. Appl. Artif. Intell.* **133**, 108270 (2024)
15. Dai, J., Upadhyay, S., Aivodji, U., Bach, S.H., Lakkaraju, H.: Fairness via explanation quality: evaluating disparities in the quality of post hoc explanations. In: *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 203–214 (2022)
16. Mohseni, S., Zarei, N., Ragan, E.D.: A multidisciplinary survey and framework for design and evaluation of explainable AI systems. *ACM Trans. Interact. Intell. Syst. (TiIS)* **11**(3–4), 1–45 (2021)
17. Arrieta, A.B., et al.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion* **58**, 82–115 (2020)
18. Polato, I., Ré, R., Goldman, A., Kon, F.: A comprehensive view of hadoop research—a systematic literature review. *J. Netw. Comput. Appl.* **46**, 1–25 (2014)
19. Abualigah, L., Masri, B.A.: Advances in mapreduce big data processing: platform, tools, and algorithms. In: *Artificial Intelligence and IoT: Smart Convergence for Eco-Friendly Topography*, pp. 105–128 (2021)
20. Silaparasetty, N.: The tensorflow machine learning library. In: *Machine Learning Concepts with Python and the Jupyter Notebook Environment: Using Tensorflow 2.0*, pp. 149–171 (2020)
21. Hu, H., Wen, Y., Chua, T.-S., Li, X.: Toward scalable systems for big data analytics: a technology tutorial. *IEEE Access* **2**, 652–687 (2014)
22. Price II, W.N., Cohen, I.G.: Privacy in the age of medical big data. *Nat. Med.* **25**, 37–43 (2019)
23. Abouelmehdi, K., Beni-Hessane, A., Khaloufi, H.: Big healthcare data: preserving security and privacy. *J. Big Data* **5**(1), 1–18 (2018). <https://doi.org/10.1186/s40537-017-0110-7>
24. Jia, Z., Zaharia, M., Aiken, A.: Beyond data and model parallelism for deep neural networks. *Proc. Mach. Learn. Syst.* **1**, 1–13 (2019)
25. Rai, A.: Explainable AI: from black box to glass box. *J. Acad. Mark. Sci.* **48**, 137–141 (2019)
26. He, J., Hao, Y., Wang, X.: An interpretable aid decision-making model for flag state control ship detention based on SMOTE and XGBoost. *J. Mar. Sci. Eng.* **9**(2), 156 (2021)
27. Khattar, N., Sidhu, J., Singh, J.: Toward energy-efficient cloud computing: a survey of dynamic power management and heuristics-based optimization techniques. *J. Supercomput.* **75**(8), 4750–4810 (2019). <https://doi.org/10.1007/s11227-019-02764-2>
28. Al-Jumaili, A., Muniyandi, R.C., Hasan, M.K., Paw, J., Singh, M.J.: Big data analytics using cloud computing based frameworks for power management systems: Status, constraints, and future recommendations. *Sensors* **23**(6), 2952 (2023)
29. Pasupulati, R.P., Shropshire, J.: Analysis of centralized and decentralized cloud architectures. In: *SoutheastCon 2016*, pp. 1–7. IEEE (2016)

30. Lim, W., et al.: Federated learning in mobile edge networks: a comprehensive survey. *IEEE Commun. Surv. Tutor.* **22**(3), 2031–2063 (2020)
31. Aldughayfiq, B., Ashfaq, F., Jhanjhi, N.Z., Humayun, M.: Explainable AI for retinoblastoma diagnosis: interpreting deep learning models with lime and shap. *Diagnostics* **13**(11) (2023)
32. Ayinde, B.O., Inanc, T., Zurada, J.M.: Redundant feature pruning for accelerated inference in deep neural networks. *Neural Netw.* **118**, 148–158 (2019)
33. Peng, P., You, M., Xu, W., Li, J.: Fully integer-based quantization for mobile convolutional neural network inference. *Neurocomputing* **432**, 194–205 (2021)
34. Elshaw, R., Sakr, S., Talia, D., Trunfio, P.: Big data systems meet machine learning challenges: towards big data science as a service. *Big Data Res.* **14**, 1–11 (2018)
35. Chen, W., Milosevic, Z., Rabhi, F.A., Berry, A.: Real-time analytics: concepts, architectures and ML/AI considerations. *IEEE Access* (2023)
36. Blasch, E., et al.: Machine learning/artificial intelligence for sensor data fusion-opportunities and challenges. *IEEE Aerosp. Electron. Syst. Mag.* **36**(7), 80–93 (2021)



An Ensemble of Yolov5 Models in Real-Time Object Detection in Low Resource Settings

Nkalubo Lenard Byenkya¹ , Nakibuule Rose², and Okila Nixon³

¹ Department of Data Science and Artificial Intelligence, Kyambogo University, Kampala, Uganda

lenkalubo@gmail.com

² Department of Computer Science, Makerere University, Kampala, Uganda
rnakibuule@cit.ac.ug

³ Department of Computer Science, Lira University, Lira, Uganda

Abstract. Object detection is a fundamental task in computer vision that aims to locate and classify objects within an image or video. It plays a vital role in various applications, including autonomous driving, surveillance systems, and robotics. One-stage object detectors such as YOLO models are very fast but they are also less accurate. To improve the performance of YOLO models, the objective of this study is to develop an ensemble model of YOLOv5 models. By combining the outputs of multiple base models, ensemble methods can achieve better accuracy and robustness. The base models used for our proposed ensemble model are YOLOv5n, YOLOv5s, and YOLOv5m. The performance metrics used are mAP, precision, and recall. The experimental results revealed that the proposed ensemble model had mAP is 56.1%, precision of 69.7%, and recall of 42.5%. In relation to the base models used in this study, YOLOv5n had mAP of 48.3% precision of 57.5%, and recall of 37.4%; YOLOv5s had mAP of 53.2%, precision of 64.7%, and recall of 39.1%; and YOLOv5m had mAP of 54.8%, precision of 66.2% and recall of 41.6%. In conclusion, our proposed ensemble model performed well when compared to the base models. The proposed model was able to detect bad roads (pothole roads) that are common in Uganda. Finally, we intend to further our research by integrating the proposed model with continual learning approaches so that the model can improve its detections with time.

Keywords: Object detection · YOLOv5 · Ensemble learning

1 Introduction

Object detection is a fundamental task in computer vision that aims to locate and classify objects within an image or video [1]. It plays a vital role in various applications, including autonomous driving, surveillance systems, and robotics [2]. Over the years, significant advancements have been made in object detection techniques, leading to the development of several successful models [3].

One-stage and two-stage object detection models are two prominent approaches in the field [4]. One-stage models, such as You Only Look Once (YOLO) and Single Shot MultiBox Detector (SSD), directly predict object bounding boxes and class probabilities in a single pass. These models are known for their real-time performance, making them suitable for applications that require fast and efficient object detection [3, 4]. On the other hand, two-stage models, such as Region-based Convolutional Neural Networks (Faster R-CNN) and Mask R-CNN, employ a two-step process [2]. The first step generates region proposals, and the second step classifies and refines these proposals. Two-stage models are known for their higher accuracy but often require more computational resources and time [5].

Ensemble methods have been widely explored in the field of object detection to leverage the strengths of different models and improve overall performance [6]. By combining the outputs of multiple base models, ensemble methods can achieve better accuracy and robustness. Ensemble modeling is the process of developing several distinct models to forecast a result, either by utilizing a variety of training data sets or a wide range of modeling algorithms [7, 8]. After that, the ensemble model adds up all of the base model predictions to provide a single, final forecast for the unknown data. Reducing forecast generalization error is the driving force behind the use of ensemble models. When using the ensemble approach, the model's prediction error lowers as long as the base models are independent and diversified. When producing a prediction, the method looks to the wisdom of the crowds.

The ensemble model behaves and functions as a single model even when it has several basis models [9]. YOLOv5 is the fastest and greatest real-time object identification method available [10, 11]. The main applications of the YOLO classifiers family in computer vision are offline and real-time object identification and image classification. One-stage object detectors such as YOLO models are very fast but they are also less accurate [12, 13]. To improve the performance of YOLO models, the objective of this study is to develop an ensemble model of YOLOv5 models. By combining the outputs of multiple base models, ensemble methods can achieve better accuracy and robustness.

The contributions of this study are multifaceted and address key challenges in real-time object detection for low-resource settings. First, the study proposes a novel ensemble model combining YOLOv5n, YOLOv5s, and YOLOv5m to improve detection accuracy and robustness through majority voting. Second, it introduces a localized dataset that captures diverse environmental conditions and object classes specific to Uganda, enhancing the model's relevance and applicability. Third, the study extensively evaluates the proposed ensemble against baseline YOLOv5 models and other state-of-the-art approaches, demonstrating superior performance in terms of mAP, precision, and recall. Finally, by tailoring the model for low-resource settings and proposing integration with continual learning for future adaptability, the study contributes a scalable and practical solution to real-time object detection challenges in constrained environments. These contributions collectively advance the field and provide a strong foundation for further research.

The main contributions of this study are:

- i. an ensemble model that utilizes YOLOv5 variants for object detection suitable for local settings, and

- ii. a novel dataset that was used for fine-tuning the proposed model to local settings, especially developing countries.

2 Related Work

The study [14] proposed an ensemble of an Ensemble Weight-Assisted YOLOv5-Based Deep Learning Technique for the Localization and Detection of Malaria Parasites. Experimental results revealed that the model YOLOv5 with the ensemble weights yields better results in terms of precision, recall, and mAP values: 0.76, 0.78, and 0.79, respectively. The model performed better than the Faster RCNN and SSD net.

The study [11] proposed an improved YOLOv5 Network for Real-Time Object Detection in Vehicle-Mounted Camera Capture Scenarios. The modified YOLOv5 network outperformed the baseline model (YOLOv5s) in terms of mean average accuracy (mAP).

The study [15] proposed an Enhanced YOLOv5 Object Detection Algorithm for the Accurate Detection of Adult *Rhynchophorus ferrugineus*. According to the results, the improved YOLOv5 algorithm performs significantly better than related models, achieving an average precision of 90.1% (mAP@0.5) and a precision of 93.8%.

The study [16] proposed a novel method based on the fine-tuning approach to enhance CNN-based deep learning models' functionality. Experiments were conducted using actual data from the province of Kayseri. Classic Faster R-CNN, MobileNet, YOLOv4, YOLOv5, and YOLOv7 algorithms produce f1 scores of 0.763, 0.677, 0.880, 0.943 and 0.842, respectively. YOLOv4, YOLOv5, YOLOv7, MobileNet, and the suggested fine-tuned Faster R-CNN techniques obtained f1 scores of 0.845, 0.775, 0.932, 0.972, and 0.889, in that order.

The study [17] proposed a fast and accurate object detector for autonomous driving based on improved YOLOv5. By training-inference decoupling, the YOLOv5 algorithm is enhanced by the use of structural re-parameterization, which increases the model's accuracy and speed. Furthermore, during the training phase, the multi-branch re-parameterization module's redundant branches are removed using the neural architecture search approach, improving the accuracy and efficiency of training. Ultimately, the network gains a tiny object detection layer and all detection layers receive the coordinate attention mechanism, which increases the model's recognition rate of walkers and small cars. The experimental results demonstrate that the suggested method outperforms many current mainstream algorithms with a detection accuracy of 96.1% on the KITTI dataset and an FPS of 202.

The study [18] proposed an improved method MSS-YOLOv5 for object detection. According to MSS-YOLOv5s experimental results, the method increases mAP by 2.4% and 2.9%, on the PASCAL VOC2007 and PASCAL 2012 datasets respectively. It also keeps up a quick inference speed.

The study [19] proposed an improvement of the YOLOv5 Model in the Optimization of the Brown Spot Disease Recognition Algorithm of Kidney Bean. According to the findings, the improved YOLOv5 model has mean average precision (mAP@0.5), recall, and detection precision of 94.7%, 88.2%, and 92.5%, respectively. These values are 4.9%, 0.5%, and 25.6% higher, respectively, than the original YOLOv5 model. The improvements in detection Precision were 1.8%, 3.0%, 9.4%, and 9.5%, respectively,

as compared to the YOLOv5-SE, YOLOv5-BiFPN, FasterR-CNN, and EfficientDet models.

The study [20] proposed an improved YOLOv5-Based Lightweight Object Detection Algorithm for People with Visual Impairment to Detect Buses. Based on the experimental findings, the mAP@0.5, recall, and precision of the Improved-YOLOv5 are better than those of the original YOLOv5. Subsequent examination of the model complexity indicates that the Improved-YOLOv5 is more efficient since it uses less memory, has fewer FLOPS, and fewer parameters, and can infer information more quickly.

The study [21] proposed an improved YOLOv5 Model with Application to Mixed impurities Detection for Walnut Kernels. Based on the findings, the improved YOLOv5 model's mean average precision (mAP) is 88.9%, which is greater than other detection networks and 6.7% higher than the original YOLOv5 network's average accuracy. Additionally, just 3.9% less detections of minor contaminants are made by the improved YOLOv5 model compared to the original YOLOv5 network. Based on the related work from the literature, this study also proposes to improve on the YOLOv5 models by developing an ensemble model based on three YOLOv5 base models i.e. YOLOv5n, YOLOv5s, and YOLOv5m. It is hypothesized that the proposed model performs better than the base models.

Ren et al. [11] proposed an improved YOLOv5 network optimized for real-time applications, achieving superior mean average precision (mAP) in vehicle-mounted scenarios. Similarly, Wu et al. [15] enhanced YOLOv5 for detecting *Rhynchophorus ferrugineus*, demonstrating precision improvements in specific object detection tasks. He et al. [18] introduced MSS-YOLOv5, focusing on balancing speed and accuracy for efficient object detection.

Additionally, Yu et al. [21] developed a lightweight YOLOv5 model for detecting impurities in food products, showcasing its applicability in constrained environments. These studies highlight the versatility and adaptability of YOLOv5, forming the basis for its integration into ensemble models like the one proposed in this work.

In addition to these advancements, the development of YOLOv10 [22] has further pushed the boundaries of object detection. YOLOv10 introduces novel architectural changes and optimization techniques that enhance detection accuracy and computational efficiency, making it suitable for a broader range of applications. The integration of YOLOv10 into ensemble models holds promise for achieving even higher performance in complex detection tasks.

3 Materials and Methods

3.1 Dataset Description

In this study, we used datasets collected around Kampala, Uganda. The image data were collected in the morning, afternoon, and evening, during both dry and rainy environments, both indoor and outdoor, and since we plan to deploy our model to our local setting to help people with blind impairments in their movements, the collected images were from an environment where blind people move, and interact with their peers and families around Kyambogo University. We collected 1000 RGB images with 20 classes (pothole_road, dish, dog, donkey, door, fence, flowers, grass, person, pothole, sign_post, stairs, stones,

tarmac, trees, bicycle, Boda, building, car, and walk_way). The images were collected using an ordinary smartphone with the specifications: Infinix Hot 40i (RAM: 8 + 8 GB, ROM: 256 GB), 6.6" HD + Display, 50 MP Dual AI Rear Camera, 32 MP Crystal-Clear Front Camera, 5000 mAh Battery With 18 W Fast Charge. We applied preprocessing and argumentation techniques before the dataset was split into training, testing, and validation. During data processing, we resized the dataset into 640×640 which is the standard size for yoloV5, after which we applied auto orientation. For data augmentation, the following operations were performed: Horizontal flip, rotation of 90 degrees both clockwise and counter-clockwise, rotation between -15 degrees and 15 degrees, shear of ± 10 horizontal and vertical, grayscale to 15% of the images, Hue between -15 degrees and 15 degrees, saturation between -25 degrees and 25 degrees, brightness between -15 degrees and 15 degrees, exposure between -10 degrees and 10 degrees, blur of up to 2.5 px and noise of up to 0.1% of pixels. The data was then split as the following: 70% for training, 20% for validation, and 10% for testing. Figure 1 shows some of the images we collected.



Fig. 1. The sample of the Image dataset used during this study.

3.2 The Proposed Model

3.2.1 The YOLOv5 Architecture

YOLOv5 is a powerful object detection algorithm built by Ultralytics. The YOLOv5 is the fifth version of the ground-breaking “You Only Look Once” [1] object recognition model designed to provide real-time, high-speed, high-accuracy findings. Constructed upon PyTorch, this potent Deep-learning framework has gained considerable recognition because of its adaptability, user-friendliness, and superior performance. YOLOv5 has been designed to be super easy and simple to learn with great performance. Figure 2

shows the comparative performance of YOLOv5 models on the COCO dataset. The architecture of YOLOv5 is divided into three primary sections: 1) Backbone: The network’s central component. The New CSP-Darknet53 structure [23], a revision of the Darknet architecture used in earlier iterations, is utilized to create the backbone of YOLOv5. 2) Neck: The neck is the structure that joins the head to the backbone. SPPF and New CSP-PAN structures are used in YOLOv5. 3) Head: This section is in charge of producing the finished product. For this, YOLOv5 employs the YOLOv3 Head. In YOLOv5, the activation functions Sigmoid [24] and Leaky ReLU [25] are used to balance the recall and precision values. The Leaky ReLU is utilized in the middle levels of the network in the real model, whereas the Sigmoid is used in the prediction layer, which is the final layer of the model. YOLOv5 differs somewhat from its predecessors in the following ways: 1) a 6x6 Conv2d structure takes the role of the Focus structure present in previous iterations. 2) SPPF takes the place of the SPP structure. Processing speed is more than doubled by this modification. YOLOv5 have several different variants which includes the following: YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x, YOLOv5n6, YOLOv5s6, YOLOv5m6, YOLOv5l6, and YOLOv5x6 + TTA. Table 1 below shows the pre-trained checkpoints of the various variants of YOLOv5 models.

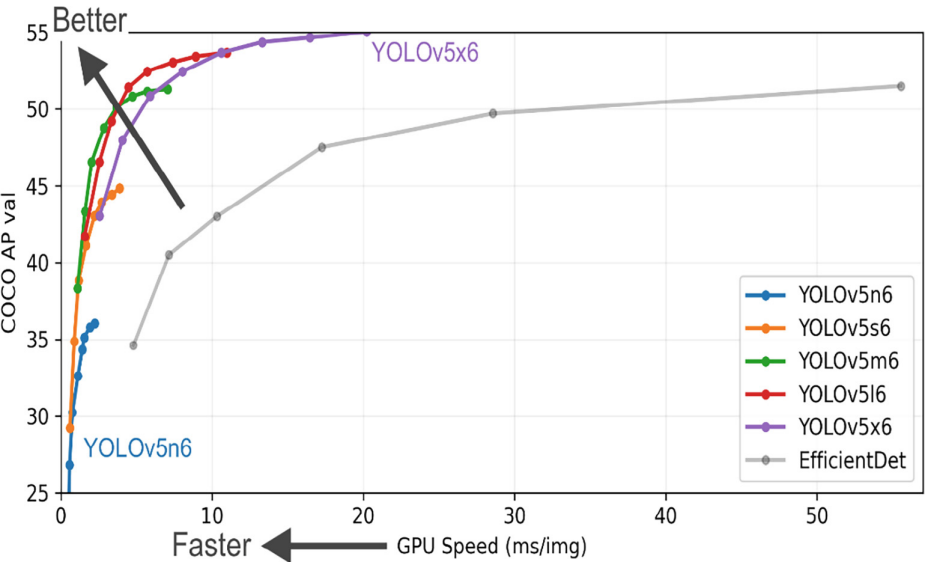


Table 1. YOLOv5 Pre-trained Checkpoints

Model	Size (pixels)	mAP ^{val} 50–95	mAP ^{val} 50	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	Params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	45	6.3	0.6	1.9	4.5
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7
YOLOv5n6	1280	36.0	54.4	153	8.1	2.1	3.2	4.6
YOLOv5s6	1280	44.8	63.7	385	8.2	3.6	12.6	16.8
YOLOv5m6	1280	51.3	69.3	887	11.1	6.8	35.7	50.0
YOLOv5l6	1280	53.7	71.3	1784	15.8	10.5	76.8	111.4
YOLOv5x6	1280	55.0	72.7	3136	26.2	19.4	140.7	209.8
+ TTA	1536	55.8	72.7	-	-	-	-	-

3.2.2 The Loss Function of YOLOv5

In YOLOv5, the loss is calculated by squaring the difference between the ground truth and the predictions. The loss is the sum of three separate loss components according to the study [1]: 1) the classification loss (BCE Loss): This is the Binary Cross-Entropy loss, which calculates the classification task's error. 2) the localization loss (errors between the ground truth and the anticipated border box). 3) The confidence loss (the box's objectivity).

Classification loss (L_{cls})

If an object is detected, the squared error of the class conditional probability for each class determines the classification loss for each cell. The classification loss is shown in Eq. 1 below.

$$L_{cls} = \sum_{i=0}^{s^2} \mathbb{I}_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \quad (1)$$

where,

$\mathbb{I}_i^{obj} = 1$ if an object appears in a cell i , otherwise 0.

$\hat{p}_i(c)$ denotes the conditional class probability of class c in cell i .

Localization loss (L_{loc})

The localization loss quantifies the inaccuracies in the anticipated bounding box dimensions and placements. We count only the box that is in charge of object detection. The

localization loss is shown in Eq. 2 below.

$$L_{loc} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \quad (2)$$

where

$\mathbb{I}_i^{obj} = 1$ if the j th boundary box in cell i is responsible for detecting the object, otherwise 0.

λ_{coord} increase the weight for the loss in the boundary box coordinates.

Absolute errors in both large and small boxes shouldn't be given the same weight. In other words, a 2-pixel inaccuracy in a big box equals a 2-pixel error in a tiny box. Instead of predicting the width and height of the bounding box, YOLO predicts the square root of it, which helps to partially fix the problem. Moreover, we compound the loss by λ_{coord} (default: 5) to emphasize the border box precision even further.

Confidence loss (L_{obj})

The confidence loss (a measure of the objectness of the box) is as follows if an object is found inside: The confidence loss when the object is inside the box is shown in Eq. 3 below.

$$L_{obj-inbox} = \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{ij}^{obj} (C_i - \hat{C}_i)^2 \quad (3)$$

where,

$\hat{C}_i$ is the box confidence score of the box j in cell i .

$\mathbb{I}_i^{obj} = 1$ if the j th boundary box in cell i is responsible for detecting the object, otherwise 0.

If an object is not detected in the box, the confidence loss when the object is not detected inside the box is shown in Eq. 4 below.

$$L_{obj-notbox} = \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \quad (4)$$

where,

$\mathbb{I}_{ij}^{noobj}$ is the complement of $\mathbb{I}_{ij}^{obj}$.

$\hat{C}_i$ is the box confidence score of the box j in cell i .

λ_{noobj} weights down the loss when detecting the background.

Most boxes are empty of any objects. This leads to an imbalance in classes; that is, we train the model to identify backgrounds more often than objects. We correct this by deducting this loss by λ_{noobj} (0.5 by default).

The Overall Loss Function used in YOLOv5

The overall loss is the summation of localization, confidence and classification losses together. The overall loss for the proposed model is calculated using the Eq. 5 below.

$$LOSS = L_{cls} + L_{loc} + L_{obj-inbox} + L_{obj-notbox} \quad (5)$$

3.2.3 The Ensemble Model

In this study, we propose an ensemble model using three YOLOv5 models. The base models used are YOLOv5n, YOLOv5s, and YOLOv5m. The goal of ensemble learning is to choose a collection, or ensemble, of hypotheses, such as h_1 , h_2 , and h_n , and aggregate their predictions using methods like voting, averaging, or a higher level of machine learning [26]. In this study, we used majority voting [27] to build our ensemble. It works by first building two or more separate models from your training data. When required to make predictions for new data, a Voting Classifier can be used to wrap your models and average the predictions of the sub-models. We maintained the default parameters and their values for YOLOv5n, YOLOv5s, and YOLOv5m in all our experiments.

3.2.3.1 Summary of the Proposed Ensemble Model

Training Phase: In this phase, we identified the models that we intended to use. We identified three YOLOv5 models and these are YOLOv5n, YOLOv5s, and YOLOv5m. The selected models represent different sizes and complexities within the YOLOv5 family. This selection allows us to explore a balance between model efficiency (speed and resource usage) and accuracy, which is critical for deployment in low-resource settings.

Prediction Phase: All trained models receive the data as input when they have to predict a new data point. Every model independently predicts and the prediction made by each model is regarded as a “vote” for a specific class.

Majority Decision: The class label that gets the most votes from each of the individual models determines the final forecast. The algorithm can use a variety of tie-breaking techniques, such as choosing the first or last prediction, in the event of a tie (Fig. 3).

3.2.3.1 The Pseudocode of the Proposed Ensemble Model

Algorithm1: The Proposed Ensemble model
Input: Image data collected in the local Environment
Output: Image with boundary boxes and classes
START
Input the dataset
Preprocess the dataset
Auto orientation
Resize
Data Augmentation
Horizontal flip
Rotation: 90 degrees both clockwise and counter-clockwise
Rotation: between -15 degrees and 15 degrees
Shear: +-10 horizontal and vertical
Grayscale: 15% of the images
Hue: between -15 degrees and 15 degrees
Saturation: between -25 degrees and 25 degrees
Brightness: between -15 degrees and 15 degrees
Exposure: between -10 degrees and 10 degrees
Blur: up to 2.5px
Noise: up to 0.1% of pixels.
Data Split
Training: 70%
Validation: 20%
Testing: 10%
Download YOLOv5 models
YOLOv5n,
YOLOv5s
YOLOv5m
Build an Ensemble model
Majority voting approach
Set epoch to 0
While epoch is < 250
Train the ensemble model
Validate the ensemble model
End While loop
Test the trained model with the test dataset
END

Fig. 3. The Pseudocode of the proposed model

3.3 Performance Metrics Used

The proposed model was evaluated using three performance metrics. These are: Mean average precision (mAP), precision, and Recall as shown in Eqs. 6–8 respectively. Mean average precision (mAP): mAP is the average of AP scores over a set of object categories. It is a more comprehensive measure of accuracy than average precision (AP), as it takes into account the performance of the model on all object categories. Precision measures the accuracy of positive predictions. A high precision means the model identifies the right objects most of the time, minimizing false positives. Recall measures the completeness of positive predictions. A high recall means the model misses very few real objects, minimizing false negatives. Also, loss functions discussed in the Eqs. 1–5 were used.

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (6)$$

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

where in mAP , N is the number of queries and AP is the Average Precision. In Eqs. 7 and 8, TP , TN , FP , and FN stand for true positive, false positive, and negative, respectively.

3.4 Model Implementation

During this study, all the experiments were done in Google Colab¹. Also, a Windows 10 laptop, Intel Core i7 processor of 2.50 GHz, and 16GB RAM with 2GB of NVIDIA GeForce MX150 graphical processing unit (GPU) was used. The default settings for YOLOv5n, YOLOv5s, and YOLOv5m were retained during the experiments. Also, all the experiments used 250 epochs.

4 Experimental Results

4.1 Results on Training and Validation Loss

The results in Fig. 4 show the performance of the proposed ensemble model on training (first row) and validation loss (second row) in relation to confidence loss, classification loss, and localization loss respectively. The results on the training loss reveal that all three loss functions reduced with the increase in the number of epochs.

4.2 Results on Box, Class, and Object Losses

The results in Fig. 5 show the overall performance of the proposed ensemble model in relation to localization loss (box loss), classification loss (class loss), and confidence loss (object loss), from left to right respectively. The results revealed that the class loss decreased greatly with the increase in the number of epochs. However, the box loss and object loss were slightly reduced with the increase in the number of epochs.

¹ https://colab.research.google.com/?utm_source=scs-index.

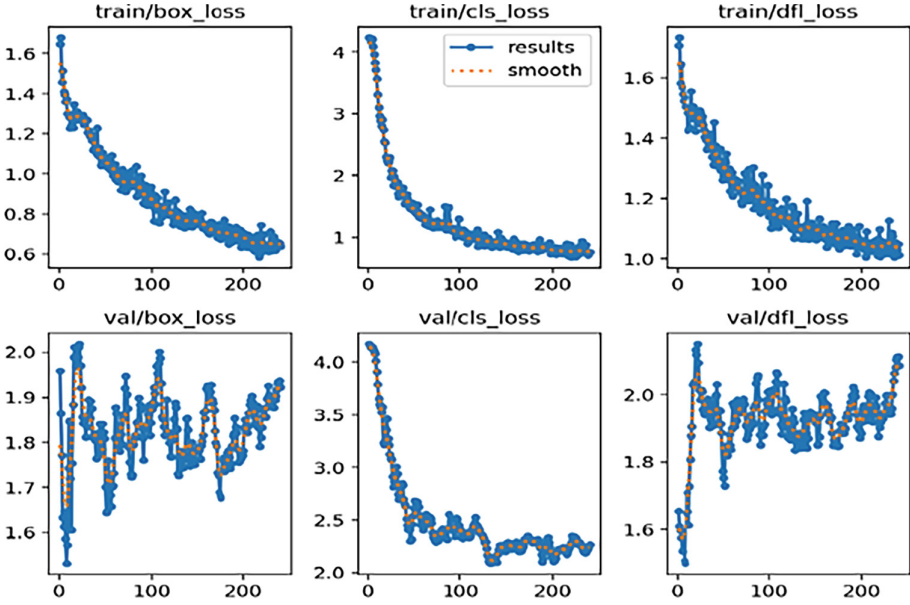


Fig. 4. Results on Training and Validation Loss

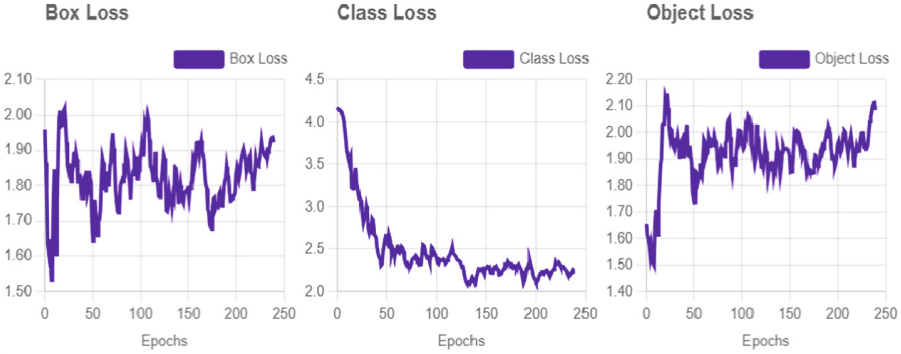


Fig. 5. Results on Box, Class, and Object Losses

4.3 Results on mAP, Precision and Recall

The results in Fig. 6 show the performance of the proposed ensemble model on precision and recall (first row), and mAP (second row). The results reveal that the precision, recall, and mAP increased with the increase in the number of epochs. The results further show that the proposed ensemble model had mAP is 56.1%, precision of 69.7%, and recall of 42.5%. In relation to the base models used in this study, YOLOv5n had mAP of 48.3% precision of 57.5%, and recall of 37.4%; YOLOv5s had mAP of 53.2%, precision of 64.7%, and recall of 39.1%; and YOLOv5m had mAP of 54.8%, precision of 66.2% and

recall of 41.6%. This shows that our proposed ensemble model performed well when compared to the base models.

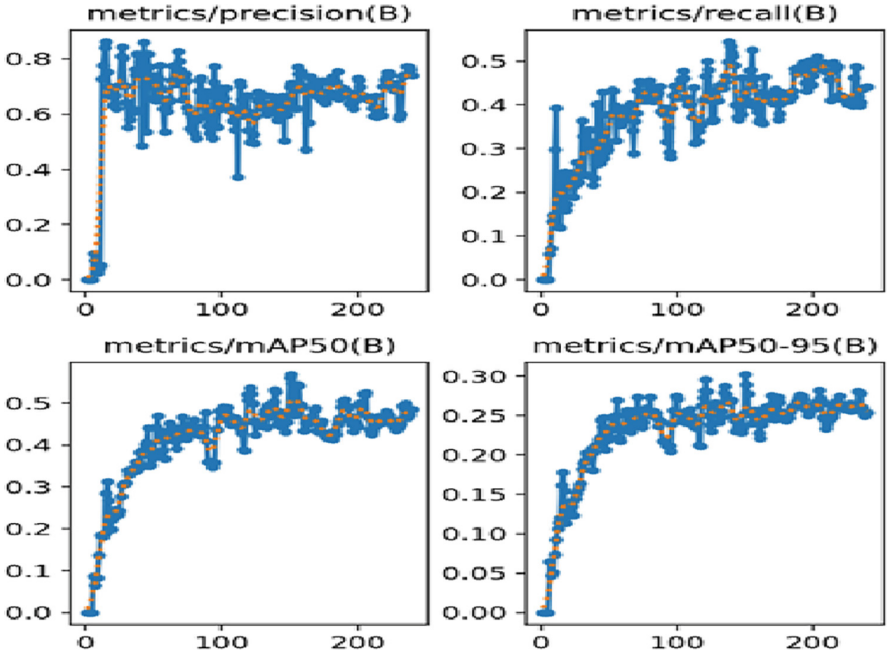


Fig. 6. Results on mAP, Precision and Recall

4.4 Results on Average Precision by Class

The results in Fig. 7 show the average precision on validation (left) and test (right). The validation results on the average precision of all classes was 56%, with boda, building, and fence classes having an average precision of 100%, while bicycle and tarmac classes had an average precision of 0%. The test results on the average precision of all classes was 69%, with building, fence, and tarmac classes having an average precision of 100%, while the walk_way class had the lowest average precision of 14%.

4.5 Results on Object Detections

The results in Fig. 8 show results on the detected objects. The proposed ensemble model was fine-tuned for object detection in the local environment in Uganda. This is because we plan to deploy our model to our local setting to help people with blind impairments in their movements, the collected images were from an environment where blind people move and interact with their peers and families around Kyambogo University. The results in Fig. 8 reveal that our proposed model was able to detect bad roads (pothole roads) that are common in Uganda. The model also detected objects such as walkways, buildings, trees, stones, and other objects mentioned in Sect. 3.1.

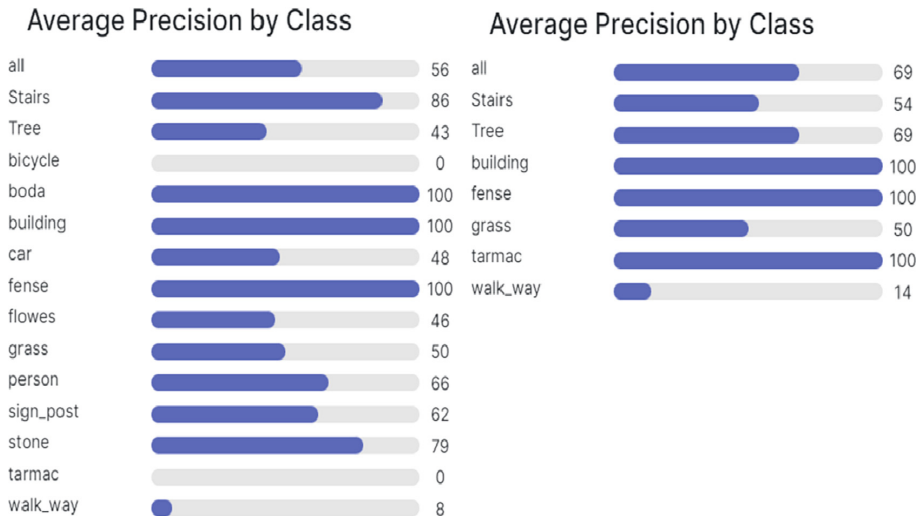


Fig. 7. Results on Average Precision by Class

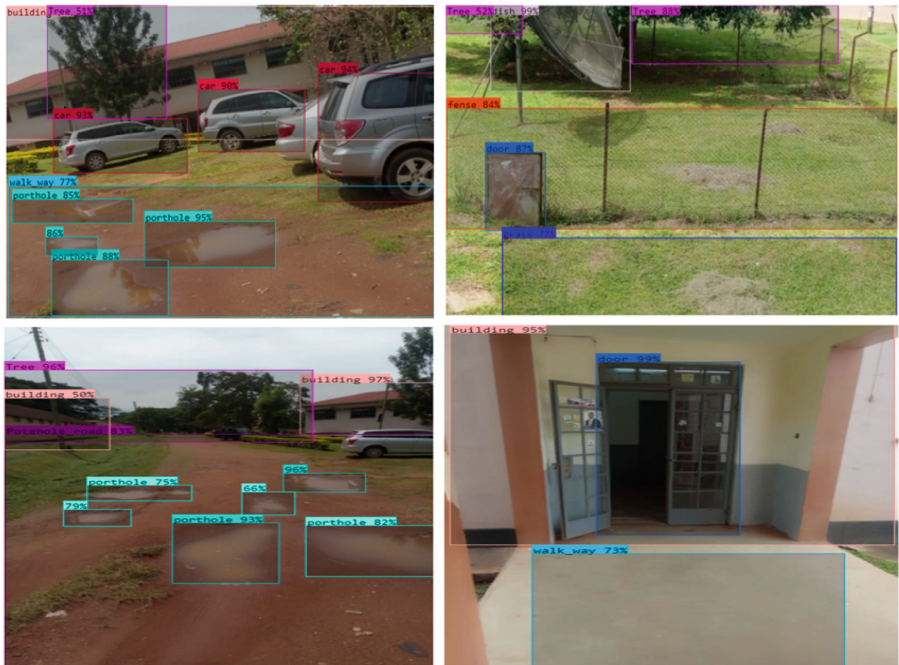


Fig. 8. Results on Object Detections

4.6 Comparison of the Proposed Model with the Current State of Art Models

Table 2 compares the performance of the proposed ensemble YOLOv5 model with state-of-the-art models from the literature. The evaluation includes datasets such as COCO, PASCAL VOC, and various custom datasets, highlighting key metrics like mAP, precision, recall, and FPS. The proposed model, tailored for a low-resource setting in Kampala, achieves competitive performance with a mAP of 56.1%, precision of 69.7%, and recall of 42.5%, outperforming its base models (YOLOv5n, YOLOv5s, YOLOv5m) and demonstrating its effectiveness in local object detection scenarios.

Table 2. Comparison of the proposed model with the state of the art

Study	Dataset	Model/Approach	mAP (%)	Precision (%)	Recall (%)
[14]	Custom Malaria Dataset	Ensemble Weight-Assisted YOLOv5	79.0	76.0	78.0
[15]	Custom Dataset	Enhanced YOLOv5	90.1	93.8	N/A
[16]	Kayseri Dataset	Fine-tuned Faster R-CNN, YOLOv5, and YOLOv7	77.5	N/A	N/A
[17]	KITTI Dataset	Improved YOLOv5 with re-parameterization	96.1	N/A	N/A
[18]	PASCAL VOC2007/2012	MSS-YOLOv5	56.0	N/A	N/A
[19]	Custom Kidney Bean Dataset	Optimized YOLOv5 for Brown Spot Detection	94.7	92.5	88.2
[20]	Custom Dataset	Lightweight YOLOv5	N/A	N/A	N/A
[21]	Walnut Kernel Dataset	Improved YOLOv5 for impurity detection	88.9	N/A	N/A
Proposed Model	Custom Dataset (Kampala)	Ensemble of YOLOv5n, YOLOv5s, YOLOv5m	56.1	69.7	42.5

4.7 Ablation Study

The ablation study in Table 3 reveals that the ensemble approach outperforms individual YOLOv5 models, with improvements of 7.8% mAP compared to YOLOv5n and 1.3% mAP compared to YOLOv5m. Additionally, incorporating the majority voting mechanism enhances precision and recall, demonstrating the value of combining predictions

from diverse base models. Preprocessing techniques like augmentation and resizing also contributed to better generalization and robustness, as shown in experiments with and without these steps.

Table 3. The Ablation Study

Configuration	mAP (%)	Precision (%)	Recall (%)
YOLOv5n	48.3	57.5	37.4
YOLOv5s	53.2	64.7	39.1
YOLOv5m	54.8	66.2	41.6
Ensemble without Voting	55.4	67.8	41.8
Proposed Ensemble with Voting	56.1	69.7	42.5

5 Discussion

The proposed ensemble model was fine-tuned for object detection in the local environment in Uganda. This is because we plan to deploy our model to our local setting to help people with blind impairments in their movements, the collected images were from an environment where blind people move and interact with their peers and families around Kyambogo University. The proposed ensemble model performed better than the three base models (YOLOv5n, YOLOv5s, and YOLOv5m) in terms of mAP, precision, and recall. The experimental results revealed that the proposed ensemble model had mAP is 56.1%, precision of 69.7%, and recall of 42.5%. In relation to the base models used in this study, YOLOv5n had mAP of 48.3% precision of 57.5%, and recall of 37.4%; YOLOv5s had mAP of 53.2%, precision of 64.7%, and recall of 39.1%; and YOLOv5m had mAP of 54.8%, precision of 66.2% and recall of 41.6%. This shows that our proposed ensemble model performed well when compared to the base models. The results were supported by the study [14] that proposed an Ensemble Weight-Assisted Yolov5-Based Deep Learning Technique which performed better than Faster RCNN [5] and SSD net [28] in the detection of malaria parasites. Our results are further supported by the study [29] that revealed that their proposed ensemble model performed better obtained up to a 10% improvement from the base models. This implies that experimental results support the motivation for using ensemble models which is to reduce the generalization error of the prediction. Also, as shown in Fig. 5, the overall performance of the proposed ensemble model in relation to localization loss (box loss), classification loss (class loss), and confidence loss (object loss) was great, which implies high performance in object detection. More so, the validation results on the average precision of all classes was 56%, with boda, building and fence classes having an average precision of 100%, while bicycle and tarmac classes had an average precision of 0% with our proposed model. The test results on the average precision of all classes was 69%, with building, fence and tarmac classes having an average precision of 100%, while the walk_way class had the

lowest average precision of 14% with our proposed model. Finally, the proposed model was able to detect bad roads (pothole roads) that are common in Uganda. The model also detected objects such as walkways, buildings, trees, stones, and other objects.

6 Conclusion

In this study, we proposed an ensemble model that was fine-tuned for object detection for local settings. The base models used for our proposed ensemble model are YOLOv5n, YOLOv5s, and YOLOv5m. The goal of ensemble learning is to choose a collection of base models and aggregate their predictions. The dataset used in this study was collected in Kampala, Uganda. In this study, a majority voting approach was used. The proposed ensemble model performed better than the three base models used to develop the ensemble i.e. YOLOv5n, YOLOv5s, and YOLOv5m in terms of mAP, precision, and recall. The proposed model detected poor roads and other objects that may hinder the movement of blind persons and therefore we plan to deploy the proposed model to assist in blind movements.

6.1 Challenges and Future Work

This study faced challenges including computational complexity associated with ensemble models compared to individual base models, as well as variations in environmental conditions that impacted dataset quality, such as lighting and weather changes. Additionally, the dataset size, while sufficient for this study, could benefit from further expansion to include more diverse object classes for improved generalization. Future work will focus on integrating continual learning approaches to enable the model to adapt dynamically to new data and improve performance over time. Expanding the dataset to include additional object classes and testing the model in other low-resource settings are also planned.

References

1. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. **2016**, 779–788 (2016). <https://doi.org/10.1109/CVPR.2016.91>
2. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2017). <https://doi.org/10.1109/TPAMI.2016.2577031>
3. Liu, W., et al.: SSD Single shot multibox detector. Lect. Notes Comput. Sci. including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformat. **9905**, 21–37 (2016). https://doi.org/10.1007/978-3-319-46448-0_2
4. Redmon, J., Farhadi, A.: YOLOv3: An Incremental Improvement, (2018). <http://arxiv.org/abs/1804.02767>
5. Girshick, R.: Fast R-CNN. Proc. IEEE Int. Conf. Comput. Vis. **2015**, 1440–1448 (2015). <https://doi.org/10.1109/ICCV.2015.169>
6. Wang, X., Cai, Z., Gao, D., Vasconcelos, N.: Towards universal object detection by domain attention. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern. Recognit. **2019**, 7281–7290 (2019). <https://doi.org/10.1109/CVPR.2019.00746>

7. Ahishakiye, E., Wario, R., Mwangi, W., Taremwa, D.: Prediction of Cervical Cancer Basing on Risk Factors using Ensemble Learning. IST-Africa 2020 Conf. Proc., pp. 13 (2020)
8. Zhang, C., Ma, Y.: Ensemble Machine Learning: Methods and Applications. Springer New York Dordrecht, Heidelberg London (2012). <https://doi.org/10.1007/978-1-4419-9326-7>
9. A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, Third Edit. O'Reilly, 2023
10. Mahendrakar, T., et al.: Performance Study of YOLOv5 and Faster R-CNN for Autonomous Navigation around Non-Cooperative Targets. 2022 IEEE Aerosp. Conf. pp. 1–12 (2022)
11. Z. Ren, H. Zhang, and Z. Li, “Improved YOLOv5 Network for Real-Time Object Detection in Vehicle-Mounted Camera Capture Scenarios,” *Sensors*, vol. 23, no. 10, 2023, <https://doi.org/10.3390/s23104589>
12. Soviany, P., Ionescu, R.T.: Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction, In: Proc. - 2018 20th Int. Symp. Symb. Numer. Algorithms Sci. Comput. SYNASC 2018, pp. 209–214. (2018). <https://doi.org/10.1109/SYNASC.2018.00041>
13. Diwan, T., Anirudh, G., Tembhurne, J.V.: Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimed. Tools Appl.* **82**(6), 9243–9275 (2023). <https://doi.org/10.1007/s11042-022-13644-y>
14. Paul, S., Batra, S., Mohiuddin, K., et al.: A Novel Ensemble Weight-Assisted Yolov5-Based Deep Learning Technique for the Localization and Detection of Malaria Parasites. *Electron.* **11**(23) (2022). <https://doi.org/10.3390/electronics11233999>
15. Wu, S., et al.: Enhanced YOLOv5 Object Detection Algorithm for Accurate Detection of Adult Rhynchophorus ferrugineus, *Insects*, **14**(8), (2023). <https://doi.org/10.3390/insects14080698>
16. Taşyürek, M., Öztürk, C.: A fine-tuned YOLOv5 deep learning approach for real-time house number detection. *PeerJ Comput. Sci.* **9**, 1–26 (2023). <https://doi.org/10.7717/PEERJ-CS.1453>
17. Jia, X., Tong, Y., Qiao, H., Li, M., Tong, J., Liang, B.: Fast and accurate object detector for autonomous driving based on improved YOLOv5. *Sci. Rep.* **13**(1), 1–13 (2023). <https://doi.org/10.1038/s41598-023-36868-w>
18. He, Y., Su, Y., Wang, X., Yu, J., Luo, Y.: An improved method MSS-YOLOv5 for object detection with balancing speed-accuracy. *Front. Phys.* **10**(January), 1–13 (2023). <https://doi.org/10.3389/fphy.2022.1101923>
19. Su, P., et al.: Improvement of the YOLOv5 Model in the Optimization of the Brown Spot Disease Recognition Algorithm of Kidney Bean. *Plants*, **12**(21) (2023). <https://doi.org/10.3390/plants12213765>
20. Arifando, R., Eto, S., Wada, C.: Improved YOLOv5-Based Lightweight Object Detection Algorithm for People with Visual Impairment to Detect Buses. *Appl. Sci.* **13**(9), (2023). <https://doi.org/10.3390/app13095802>
21. Yu, L., Qian, M., Chen, Q., et al.: An Improved YOLOv5 Model: Application to Mixed Impurities Detection for Walnut Kernels. *Foods*, **12**(3), (2023). <https://doi.org/10.3390/foods12030624>
22. A. Wang *et al.*, “YOLOv10: Real-Time End-to-End Object Detection,” no. NeurIPS, pp. 1–21, 2024, [Online]. Available: <http://arxiv.org/abs/2405.14458>
23. Wang, C.Y., Mark Liao, H.Y., Wu, Y.H., et al.: CSPNet: A new backbone that can enhance learning capability of CNN. *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work* **2020**, 1571–1580 (2020). <https://doi.org/10.1109/CVPRW50498.2020.00203>
24. Nwankpa, C., Ijomah, W., Gachagan, A., et al.: Activation Functions: Comparison of trends in Practice and Research for Deep Learning, pp. 1–20. (2018). <http://arxiv.org/abs/1811.03378>
25. Xu, B., Wang, N., Chen, T., Li, M.: Empirical Evaluation of Rectified Activations in Convolutional Network. (2015). <http://arxiv.org/abs/1505.00853>

26. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 4th ed., vol. 48. Pearson Education Limited (2022). www.pearsonglobal editions.com
27. Brownlee, J.: Machine Learning Mastery With Python: Understand Your Data, Create Accurate Models and Work Projects End-To-End. (2016)
28. Srivastava, S., Divekar, A.V., Anilkumar, C., et al.: Comparative analysis of deep learning image detection algorithms. *J. Big Data* **8**(1), (2021). <https://doi.org/10.1186/s40537-021-00434-w>
29. Casado-García, Á., Heras, J.: Ensemble methods for object detection. *Front. Artif. Intell. Appl.* **325**, 2688–2695 (2020). <https://doi.org/10.3233/FAIA200407>



Hybridizing Deep Neural Networks and Machine Learning Models for Natural Scene Image Classification

Clopas Kwenda¹(✉) , Mandlenkosi Victor Gwetu^{2,3} ,
and Jean Vincent Fonou Dombeu³

¹ Faculty of Computing and Informatics, School of Informatics, Namibia University
of Science and Technology, Windhoek, Namibia

`clapas1984@gmail.com`

² Department of Industrial Engineering, University of Stellenbosch,
Stellenbosch 7600, South Africa

`mgwetu@sun.ac.za`

³ School of Mathematics, Statistics and Computer Science, University of KwaZulu
Natal, Pietermaritzburg 3209, South Africa

`FonouDombeuJ@ukzn.ac.za`

Abstract. Classifying images into interpretative classes has grown increasingly challenging. Generally, the performance of a model with regard to accuracy is primarily influenced by the trained dataset and the chosen algorithm, among other factors. This study proposes a hybridization of deep convolutional neural network and machine learning classifiers for the purpose of classifying scenery imagery into different classes of building, forest, glacier, mountain, and sea. The rationale of the hybridization approach is that CNN has the capacity to reduce the dimensionality of feature vectors by transforming them into low-dimensional representations which makes subsequent machine learning classifiers more efficient and less prone to overfitting. In this research, the ResNet50 model was employed to extract features for classifiers such as Linear Support Vector Machine (LinearSVM), Gaussian Naive Bayes (GNB), eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Light Gradient Boost Machine (LGBM). The dataset used to evaluate the model was obtained from Kaggle. The LGBM-based model performed the best as it obtained the highest classification accuracy of 0.81, followed by XGBoost, RF, LinearSVM, a fully connected ResNet50, and GNB, with classification accuracies of 0.78, 0.78, 0.69, 0.68 and 0.64, respectively. The study showed that a fully connected ResNet50 model had an inferior performance on its own than when used in conjunction with other classifiers.

Keywords: Deep learning · Machine learning · Feature Extraction · Classification

1 Introduction

The process of image classification involves assigning an image instance into its correct category, based on a set of features [1, 2]. Extraction of significant features from images is one of the primary objectives of computer vision, which has been extensively applied in face recognition [3], texture classification [4], 3D construction, age classification, and many other systems. Natural scene image classification is a particularly challenging application of computer vision as it usually obtains low classification accuracy results.

Researchers have shifted their focus to deep learning for natural scene image categorization because of its ability to learn complicated representations directly from data. This shift has opened avenues for tackling more complex tasks such as scene classification using datasets that have many scene representations [5]. However, supervised deep learning approaches require a large amount of training images, hence labeling the training images can be time-consuming.

Intending to achieve accurate and effective classification of natural scene images, this paper proposes a hybridization model of a deep neural network, ResNet50. Transfer learning is used to sequentially combine ResNet50 with several machine learning classifiers such as RF, LinearSVM, XGBoost, LGBM, and GNB that are applied on natural scene images sourced from the Kaggle data set. The effectiveness of the proposed hybrid approach for natural image classification has been evaluated by measuring classification accuracy across various class labels. These classes include forest, sea, mountain, glacier, and buildings. Experimental results demonstrate that the LGBM classifier serves as the optimal alternative for natural scene image classification.

The remainder of this paper is arranged as follows: Sect. 2 discusses related work. Section 3 gives a summary of the machine learning classifiers utilized in the study. The suggested technique is explained in Sect. 4. The methodology is outlined in Sect. 5. Section 6 concludes the study.

2 Related Studies

Image classification is a complex procedure that relies on various components [6]. Cervantes et al. [7] presented benefits and limitations of various image classification techniques such as genetic algorithms, Support Vector Machines, Neural Networks, and Fuzzy logic. This section examines how some of these image classification methods have been used in research on natural scene images.

Deep learning approaches offer considerable advantages with regard to feature extraction, which is essential for image segmentation and classification. With that in context, Szegedy et al. [8] in 2015 proposed a broader GoogleNet model based on an inception module that integrates multi-scale information for improving image classification results. Similarly, [9] developed a CNN MobileNet that is lightweight and based on deep separable convolution. The deep separable convolution aims to divide the standard convolution into several channel-specific convolutions, also known as channel convolutions. The point convolution is used

to synthesize the finished product. Although these two feature extraction techniques are thought to be effective, they were unable to obtain precise image feature information.

Kishore et al. [10] used machine learning classifiers based on graph-based techniques to classify natural images. Local features were obtained from the transition between adjacent vertices. The neighborhood property idea was used to find a dual graph function between a vertex M and its two neighbors, M_1 and M_2 . The texture image was initially divided into non-overlapping 2×2 grids, and the resulting dual graph unit (DGU) replaced the 2×2 grid. The DGTm is the dual graph texture matrix that is developed from the DGU-indexed image co-occurrence matrix. The machine learning classifier was trained using the gray level co-occurrence matrix (GLCM) features that were obtained from the DGTm as feature vectors. The proposed model outperformed other existing models based on the results of five natural scene texture images. The challenge faced by their study was obtaining the appropriate features required for image classification. The local binary descriptors adopted by their study for feature generation could not address issues to do with: (i) sensitivity to rotation, (ii) data dependence, and (iii) inappropriateness of descriptors for real-time applications where the focal point is on data rotation. To address these issues, we proposed a hybrid model based on CNN, leveraging its ability to generate features that are essential for image segmentation and classification [11].

For the categorization of natural scene images, a study by Li et al. [12] created a deep learning model based on a combination of factor analyzers and fisher vector (MFA-FVNet). The model integrates an MFA-FV layer, that utilizes the true version of MFA-FV, which is realised by coming up with optimal network combinations and computations. The MFA-FVnet is heavily reliant on more precise implementation and good statistical methods. The results obtained indicate that the MFA-FVnet has a state of the art performance on scene classification. Processing scenarios with intricate object combinations that might occur at various scales presented difficulties for their study, nevertheless. ResNet-50 addresses the challenge of complex scenes with varying object scales by introducing a deep residual learning framework. The network learns hierarchical representations, where earlier layers focus on low-level features (like edges and textures) and later layers capture more abstract, high-level features. This is the reason why our proposed study settled for the ResNet50.

Deep learning models reduce the resolution of high-resolution input images through a downsampling process, which leads to some loss of information and can impact image classification accuracy [13]. Weng et al. [13] created a deep learning model that incorporates wavelets into the image processing step to solve this problem, allowing for the extraction of all input image information while reducing information loss. The initial convolutional layers were left out and specific sub-band image channels were used as inputs for conventional CNNs. This adjustment enabled the networks to operate directly in the wavelet domain. The model improved classification by 2.15% and 10.26% on the Describable Textures data set and the Caltech-256 data set, respectively. However, because the wavelet

transform had to be applied to each image produced during the augmentation phase, the model encountered serious difficulties and high training costs. In order to address this issue, the output images from the ResNet50 architecture's third batch normalization layer were taken out, making sure the images were kept as lightly compressed as possible.

3 Overview of Machine Learning Classifiers

An overview of machine learning classifiers, including GNB, RF, LinearSVM, XGBoost, LGBM, and a Fully connected ResNet50, is given in this section.

3.1 Random Forest Algorithm

By its very nature, RF algorithm is an aggregation classifier that bases its predictions on a collection of classification and regression trees (CARTs) [14]. This approach creates trees through the use of the bagging technique. Only $\frac{2}{3}$ of the total samples—known as “in-bag samples”—are employed in tree training, the remaining $\frac{1}{3}$ known as “out-of-the-bag samples”—being used for internal cross-validation to evaluate the RF model's performance [14]. The internal cross-validation estimates the error (referred to as out-of-bag (OOB) error) produced by the model. The technique creates each decision tree on its own without any pruning, and it splits each node according to the user-specified number of features (Mtry). These features are selected randomly. The forest grows to a predetermined number of trees (Ntree), and the trees produced have a high variation and low bias. The probability values that each tree produces are averaged to make the final categorization decision. This means that a new data instance is examined using all decision trees, with each tree voting for class membership. The highest number of votes will determine the ultimate membership class. As previously explained, Ntree and Mtry are the two important parameters that are used to generate forest trees. Many researches have concluded that the Mtry contributes more to classification accuracy than Ntree [15]. Considering that the computational efficiency of RF is not much influenced by Ntree, its value is set as high as possible [16]. A great number of studies have set Ntree to 500 because that is where error stabilizes [17]. In light of this, this study decided to use Ntree to 500.

3.2 Gaussian Naive Bayes

The only classifier that does not account for covariance is Gaussian Naive Bayes (GNB). The GNB assigns every data point to a class which it is nearer to. It does this by computing the Euclidean from the class means and compares this distance to the class variance. The z-score is calculated for a particular dimension by dividing the standard deviation by the distance from the mean. Z-score distance from class A is expressed as $(x - \mu_A)/\sigma_A$, as seen in Fig. 1. This expression also caters for class B. This classifier assumes that classes follow

Gaussian normal distributions. This assumption makes it possible to transform each z-score distance into a p-value. The p-value represents the likelihood of observing a given data point. Given observed data, the Bayes concept—which is based on conditional probability—is used to calculate the likelihood of a class.

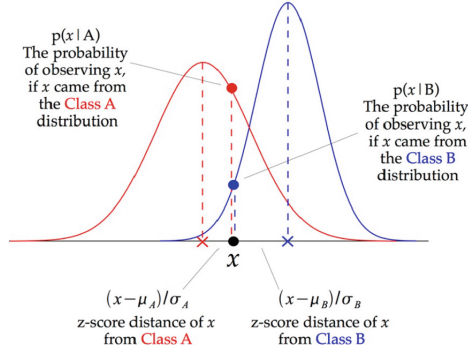


Fig. 1. Illustration of how GNB works [18].

3.3 Linear Support Vector Machine

Suppose the following data set is given $\{(m_1, n_1) \dots (m_N, n_N)\}$ the idea is to obtain a large margin classifier that separates two regions of data, where m_i and n_i represent the i^{th} data point in the feature space and its class label, respectively. A hyperplane that classifies a given set of points following the equation given in Eq. 1

$$\vec{w} \cdot \vec{m} - b = 0 \quad (1)$$

$\vec{w}$ denotes the normal vector to the hyperplane and b is the bias that shifts the plane. The hard margin strategy is used if the training data shows linear separability and is error-free, meaning it is free of noise and outliers. In situations where errors occur, it can be attributed to either a lesser margin or the failure of a hard margin. The construction of the hard margins involves the following sequential steps:

- All the points outside the margins are expressed as $W^T m_j + b \geq a$, where a is a constant related to the margin size
- Maximization of the margin is expressed as

$$Max \gamma = a ||w|| \quad (2)$$

The term $||w||$ represents the norm (magnitude) of the vector w , which defines the margin of the hyperplane. By maximizing γ , we are increasing the distance between the two classes while keeping the margin as large as possible.

- Lastly when a is set to 1, the minimum obtained is expressed as $||w|| (w^{(t)} \cdot m_j) + b \geq 1$

3.4 XGBOOST

Extreme Gradient Boosting (XGBoost) performs regression and classification tasks by utilizing an ensemble of decision trees built on the principle of gradient boosting [19, 20]. Just like gradient boosting, XGBoost constructs an additive construction of the intended objective function in a way that minimizes a loss function. Since XGBoost solely depends on decision trees as a base classifier, the complexity of trees is controlled by varying the loss function. XGboost is described as follows: Consider a set D with e samples and f features, D is expressed as:

$$D = (v_i, x_i) (|D| = e, v_i \in R^f, x_i \in R), \quad (3)$$

where v is the Eigen value, and x is the true value. The final predictive value is obtained by summing the results of all S trees. The summation process is expressed as:

$$\hat{x}_i = \sum_{k=1}^s l_s(x_i, l_s \in L), \quad (4)$$

where L is a set of decision trees and it is expressed as follows:

$$L = l(v) = h_{q(v)}(q : R^f \rightarrow Z, h \in R^Z), \quad (5)$$

where $l(v)$ represents an instance of a tree, and the weight of each node is represented by $h_{q(v)}$, q is the structure of each node and Z denotes leaf node number. Summing the leaf node values of each tree yields the XGBoost's ultimate predictive value. Because the model's goal is to learn the trees in F , the objective function is minimized as follows:

$$H^{xgb} = \sum_{i=1}^n H(p_i, \hat{p}_i) + \sum_{m=1}^m \Omega(f_k), \quad (6)$$

where H is the loss difference between predicted values $\hat{p}_i$ and actual values p_i . Overfitting of the model is avoided by setting the penalty of the decision tree with regularization Ω .

3.5 Light Gradient Boost Machine (LGBM)

Decision trees, which are poor learners, are used by the ensemble algorithms XGBoost and LGBM to identify intricate, non-linear patterns. Nevertheless, LGBM uses a tree growth based on leaves for classification and ranking. A leaf represents a type of data, and its branches represent its subgroups. Two key methods form the foundation of LGBM: exclusive feature bundling for classification and gradient-based one-sided sampling for sampling, respectively [21]. When compared to analog methods with such combined features, the processes of data scanning, classification, clustering, and sampling are completed in a shorter period and with greater accuracy. When memory requirements, arithmetic speeds, and processing time are considered, LGBM is an excellent choice.

3.6 ResNet50

Residual networks are deep CNNs that skip some block convolutional layers by creating shortcut connections [22]. The basic blocks (also referred to as bottlenecks) adopt the following two design rules:

- The resulting feature map size is directly correlated with the number of filters in the layers.
- The filter number doubles when the feature map is cut in half.

After the image is fed into ResNet50, it passes via the first convolutional layer, conv1, which has 64 filters and a 7×7 kernel size. Next, it passes through a stride-2 max-pooling layer. In the second convolutional layer, conv2_x, the layers are paired in accordance with how the residual networks are connected. The process continues up to convolutional level 5, where average pooling is applied at the fully connected layer. The ResNet50 solves the vanishing gradient problem by incorporating residual blocks that act as a shortcut connection to allow a smooth flow of information to only essential layers. The residual block comprises of three stacked layers of 1×1 , 3×3 , and 1×1 convolutions arranged in such a way that 1×1 layers are designed to reduce and restore image dimensions respectively, where the 3×3 layer creates a bottleneck with smaller input/output dimensions. The matrix in Eq. 7 shows that there are 2 layers of $kernel_{size} = 1 \times 1$ with 64 and 256 filters respectively, and another of $kernel_{size} = 3 \times 3$ with 64 filters.

$$A = \begin{pmatrix} 1 \times 1 & 64 \\ 3 \times 3 & 64 \\ 1 \times 1 & 256 \end{pmatrix} \times 3 \quad (7)$$

4 Methodology

The study introduces a hybrid of a deep neural network model and traditional machine learning approach for classifying natural scene images. This method integrates, ResNet50, with machine learning classifiers such as LinearSVM, GNB, XGBoost, RF, and LGBM. ResNet50's sole purpose in this model is to generate a feature vector that is used by the machine learning classifiers to perform subsequent image classification. These images were obtained from Kaggle [23]. The proposed workflow of the proposed model is depicted in Fig. 2. The model is based on the following properties.

- ResNet50 was employed to produce a feature vector and machine learning classifiers were used to perform classification process.
- The key advantage of ResNet50 over other neural networks is its capacity to extract relevant features from a minimal training data set.

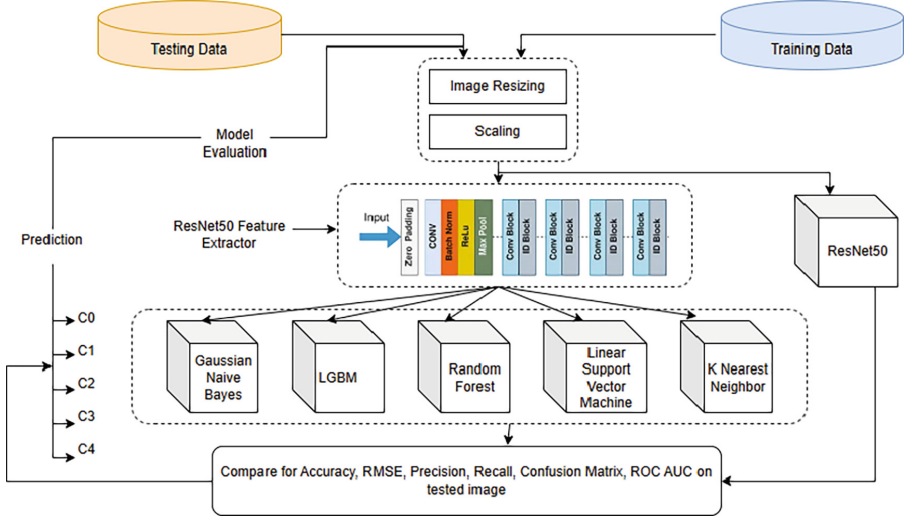


Fig. 2. The flow of the proposed model.

4.1 Multi-class Image Classification

Multi-class classification is when an instance image can assume a class from a predetermined set of classes. To enable the classification process, the transform LabelEncoder from the pre-processing library in sklearn was used to label the classes into integer discrete values. The inverse transform function was used to visualize data during the prediction phase. Each class has 22 natural scene images for training, 11 images for testing in each category, and 66 varied images for validation.

4.2 Pre-processing

Since the images in the Kaggle data set varied in size, they were all reduced to 224 by 224 pixels in order to meet the ResNet50 input size requirement. Eighty percent of the dataset was used for training, while twenty percent was used for testing. Class labels were considered categorical data, and non-numeric categories were transformed into numeric values through label encoding. The class labels, ranging from 0 to 4, were then represented as a vector. This data transformation was necessary because most machine learning algorithms require labels to be converted into numerical integer values. Table 1 represents the labeled classes. The parameters for the classifiers were configured using the random search optimization technique, as shown in Table 2. Feature scaling is a crucial step in data preprocessing for machine learning, as many models are sensitive to the scale of feature values. To ensure that the 8-bit image pixel values were normalized to a range between 0 and 1, all of the images used in this study were scaled by dividing their pixel values by 255.

Table 1. Labels of scene images

Value	Class
0	building
1	forest
2	glacier
3	mountain
4	sea

Table 2. Parameters set for the classifiers

Method	Parameters
Linear Support Vector Machine	<i>kernel</i> = 'linear', <i>class_weight</i> = 'balanced', <i>FeatureScaling</i> = Standardize, <i>param_grid</i> = 'svm.C' : [0.01, 0.1, 1, 10, 100]
Random Forest	<i>n_estimators</i> = 200, <i>max_depth</i> = 20, <i>min_samples_split</i> = 4, <i>min_samples_leaf</i> = 2, <i>max_features</i> = 'sqrt', <i>bootstrap</i> = true, <i>random_state</i> = 42
Gaussian Naive Bayes	<i>var_smoothing</i> = $1e-9$, <i>priors</i> = [0.5, 0.5], <i>random_state</i> = 42
Linear Discriminant Analysis	<i>param_grid</i> = 'solver' : ['svd', 'lsqr', 'eigen'], 'shrinkage' : ['auto', None], <i>random_state</i> = 42
Light Gradient Boost Machine	<i>learning_rate</i> = 0.1, <i>max_depth</i> = 15, <i>num_leaves</i> = 50, <i>boosting_types</i> = gbd, <i>num_iteration</i> = 1000, <i>feature_fraction</i> = 0.8, <i>max_bin</i> = 255
eXtreme Gradient Boosting	<i>learning_rate</i> = 0.3, <i>max_depth</i> = 10, <i>boosting_type</i> = gbt, <i>num_iteration</i> = 500

4.3 Evaluation Metrics for Assessing Model Performance

Several metrics were employed to assess the model's performance in multi-class classification, including accuracy, mean average error (MAE), recall, precision, confusion matrix, root mean square error (RMSE), and receiver operating characteristic - area under the curve (ROC-AUC). A forecasted image's deviance from the actual image is indicated by the Mean Average Error (MAE), which computes the average difference between expected and actual data.

5 Results and Discussion

This study seeks to identify the most effective machine learning classifier for categorizing natural images into their respective classes. This is achieved by comparing the results produced by LinearSVM, RF, GNB, LGBM, XGBoost, including a fully connected ResNet50 deep convolution neural network. Through transfer learning, ResNet50 was used to produce a set of features that the machine

Table 3. Metrics for measuring classification through RF based model

Pixel Region	Precision	Recall	F1-Score
0	0.80	0.60	0.69
1	0.83	1.00	0.91
2	0.55	0.80	0.65
3	0.91	0.80	0.65
4	0.95	1.00	0.98

Table 4. Metrics for measuring classification through XGBoost based model

Pixel Region	Precision	Recall	F1-Score
0	0.93	0.70	0.80
1	0.91	1.00	0.95
2	0.74	0.70	0.72
3	0.78	0.70	0.74
4	0.77	1.00	0.87

Table 5. Metrics for measuring classification through LGBM based model

Pixel Region	Precision	Recall	F1-Score
0	0.90	0.95	0.93
1	0.87	1.00	0.93
2	0.70	0.70	0.70
3	0.80	0.60	0.69
4	0.95	1.00	0.98

Table 6. Metrics for measuring classification through GNB based model

Pixel Region	Precision	Recall	F1-Score
0	0.55	0.85	0.67
1	0.89	0.85	0.87
2	0.50	0.35	0.41
3	0.60	0.15	0.24
4	0.65	1.00	0.78

Table 7. Metrics for measuring classification through ResNet50 model

Pixel Region	Precision	Recall	F1-Score
0	0.72	0.65	0.68
1	0.70	0.95	0.81
2	0.80	0.40	0.53
3	0.54	0.75	0.63
4	0.76	0.65	0.70

Table 8. Metrics for measuring classification through LinearSVM based model

Pixel Region	Precision	Recall	F1-Score
0	0.88	0.70	0.78
1	0.77	1.00	0.87
2	0.58	0.75	0.65
3	0.75	0.15	0.25
4	0.61	0.85	0.71

learning classifiers could use to carryout the classification process. ResNet50 is also used separately in an end-to-end manner, from feature extraction to classification.

Table 3 present results obtained by RF algorithm. In terms of precision, the algorithm was highly precise in classifying images in category 4 and performed generally poor in category 2. Precision in simple terms is the measure of quality. Precision results for categories 0, 1, and 3 were generally impressive as the average precision is above 80%. In terms of recall, the RF algorithm managed to classify all images under categories 1 and 4.

The term “measure of quantity” also applies to recall. The algorithm performed poorly for categories 0 and 3, as the average recall values swing to about

50% and 60% respectively. F1- Score is related to the accuracy, and it provides a balance between precision and recall. For category 4, the RF obtained the highest F1-Score of 98%, while category 1 had a value of 91%.

Table 4 shows results obtained by the XGBoost algorithm. In terms of precision, the algorithm was highly precise in classifying images in category 0 and immediately followed by category 1. For the 3 remaining categories, the algorithm performance is generally good as the precision values obtained are above 70%. In terms of recall, all relevant images were correctly classified under categories 1 and 4 and for the three remaining categories, each obtained a recall value of 70%. Unlike the RF, this algorithm was most accurate in classifying images under category 1. Generally in terms of F1-score the XGBoost algorithm performed well in classifying images into their respective categories as the F1-Score is generally above 75%.

As presented in Table 5 the LGBM algorithm performed very well in terms of precision, recall, and F1-Score except for category 3, which obtained the worst results. All the classifiers performed poorly for category 3 for recall and F1-score. The reason could be that some glacier images had heaped ice that looked like mountains, resulting in some misclassifications.

The recall, F1, and precision score values for the GNB classifier are presented in Table 6. With regard to precision, the model achieved the highest precision for class 0, whilst the performance for the remaining classes is marginally better as the precision values obtained revolve around 0.50 and 0.65. In terms of recall, the GNB model exhibited the worst performance in class 3 and class 2 by obtaining recall values of 0.35 and 0.15 respectively. Class 1 exhibited the highest overall performance based on the F1-score.

Performance evaluation for ResNet50 presented in Table 7 recorded the highest precision of 0.8 for class 2 and immediately followed by classes 4, 6, and 1 which achieved precision values of 0.76, 0.72, and 0.70, respectively. This indicates that the ResNet50 model effectively minimized false positives for class 1. With a recall value of 0.95, the ResNet50 network demonstrated strong performance for class 1. This finding implies that the model was highly effective in minimizing false negatives. However, it exhibited suboptimal performance in class 2, as evidenced by a recall value of 0.4. The ResNet50 model exhibited superior performance in accurately categorizing test images inside class 1, followed by classes 4, 0, 3, and 2, respectively.

As illustrated in Table 8, LinearSVM recorded the highest precision in classifying test images in class 0 by obtaining a precision value of 0.88. Precision values for class 1 and class 3 are deemed desirable due to their levels exceeding 0.7. However, the performance for class 2 and class 4 is suboptimal as the obtained values are marginally above 0.5. In terms of recall, the worst performance is recorded for class 3 with a recall value of 0.15. For the remaining classes, the overall recall performance is generally satisfactory. The overall performance in terms of F1-score is recorded in class 1.

The Gaussian Naive Bayes (GNB) model achieved absolute classification accuracy for all twenty test images in class 4, as indicated by the confusion

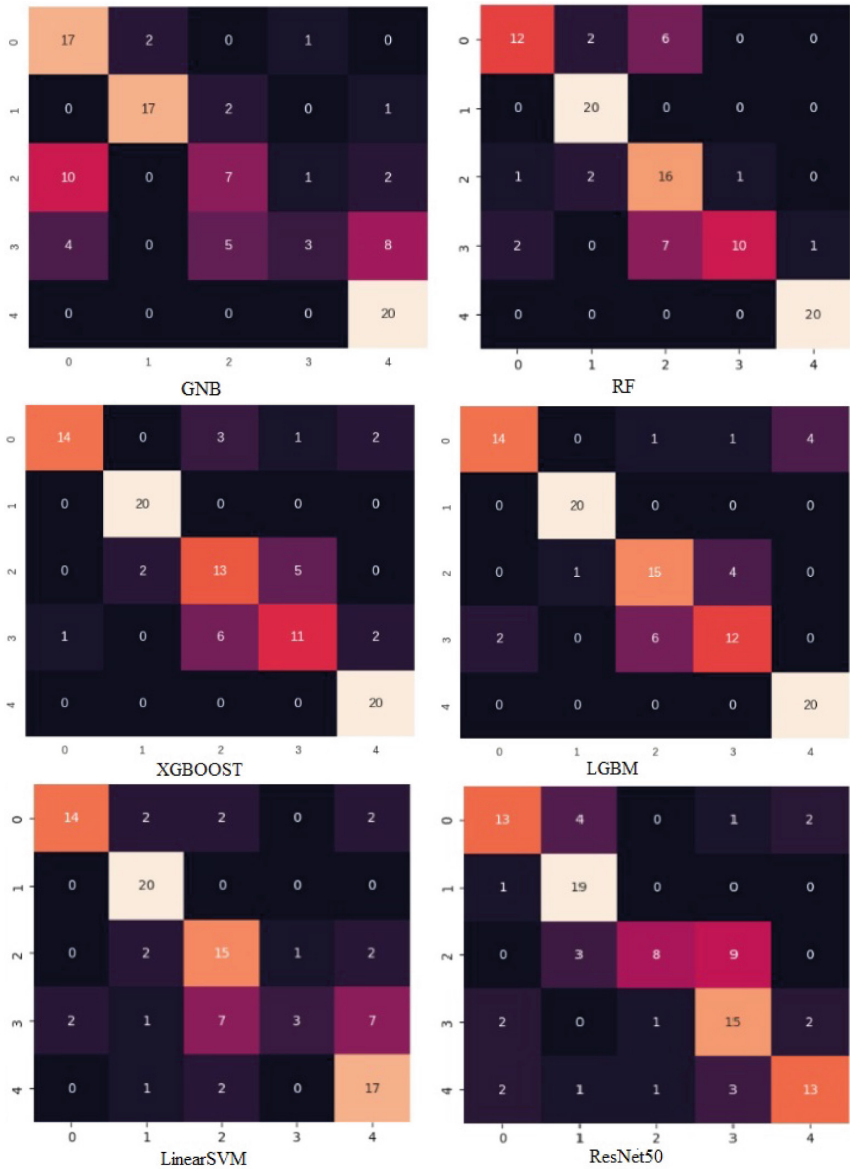


Fig. 3. Confusion Matrix for RF, XGBoost, LDA, GNB, LinearSVM, and KNN

matrix in Fig. 3. A total of seventeen test images were accurately identified for classes 0 and 1. Classes 2 and 3 exhibited the poorest performance, with just 7 and 3 test images being accurately categorized, respectively. The ROC AUC curve associated with the Gaussian Naive Bayes (GNB) classifier, as depicted in Fig. 4, aligns well with the findings presented in the confusion matrix discussed

in Fig. 3 for GNB. Class 4 achieved the highest ROC AUC of 0.93, followed by classes 1 and 0 with ROC AUC values of 0.91 and 0.84, respectively. The classes that exhibited the poorest performance were class 2 and class 3, with corresponding ROC AUC values of 0.63 and 0.59, respectively.

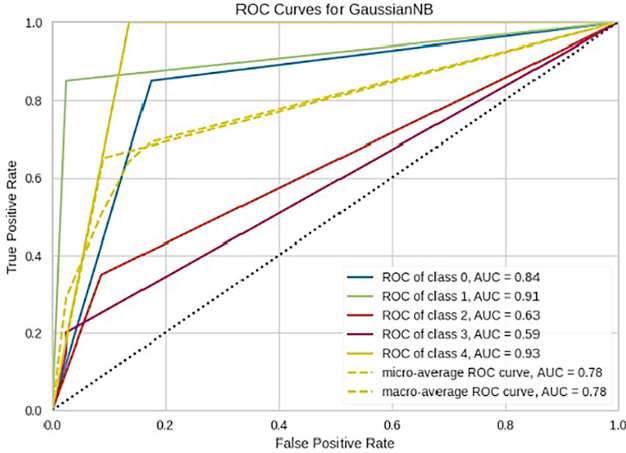


Fig. 4. ROC AUC Curve for GNB.

As depicted in Fig. 3, the RF-based classifier accurately classified all twenty test images into classes 1 and 4, with class 2 coming next, having 16 correctly classified test images. Classes 0 and 3 performed the poorest, with only 10 and 12 test images correctly classified, respectively. The ROC AUC curve associated with the RF model corresponds to the confusion matrix results for RF, as it exhibits a flawless classification performance for classes 1 and 4. The model successfully demonstrated the ability to accurately differentiate between image categories within these two classes (Fig. 5).

The confusion matrix for XGBoost in Fig. 3 presents its strength in absolutely classifying all twenty test images for classes 1 and 4. The classification of class 1 and class 4 showcases the effectiveness of the RF algorithm. This is subsequently followed by classes 0, 2, and 3 where 14, 13, and 11 test images were correctly classified. These results are also confirmed by the associated ROC AUC of XGBoost in Fig. 6. The model demonstrated a absolute classification performance for classes 1 and 4, as evidenced by a ROC AUC value of 1. Subsequently, the remaining classes, namely classes 0, 3, and 2, achieved ROC AUC values of 0.96, 0.92, and 0.91, respectively. Similar to XGBoost, both RF and LGBM correctly classified all twenty test images for classes 1 and 4 (see Fig. 3). These results align with the LGBM ROC AUC curve shown in Fig. 7. Subsequently, classes 2, 0, and 3 were accurately classified, with 15, 14, and 13 test images correctly identified, respectively. These results are consistent with the ROC AUC curve of the LGBM model, shown in Fig. 7. The results revealed that LGBM

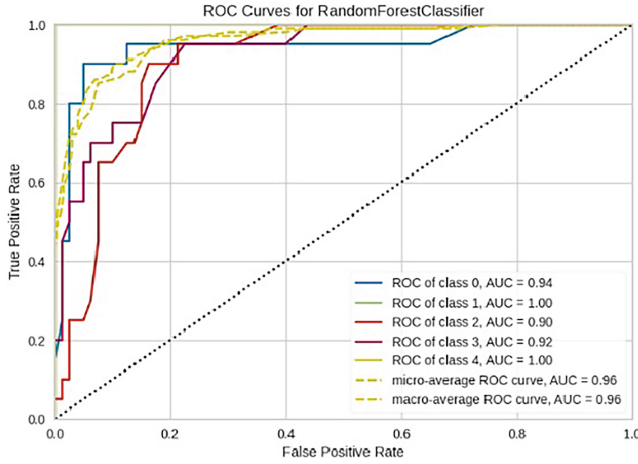


Fig. 5. ROC AUC Curve for RF.

proved to be the most effective classifier when compared to the other baseline classifiers.

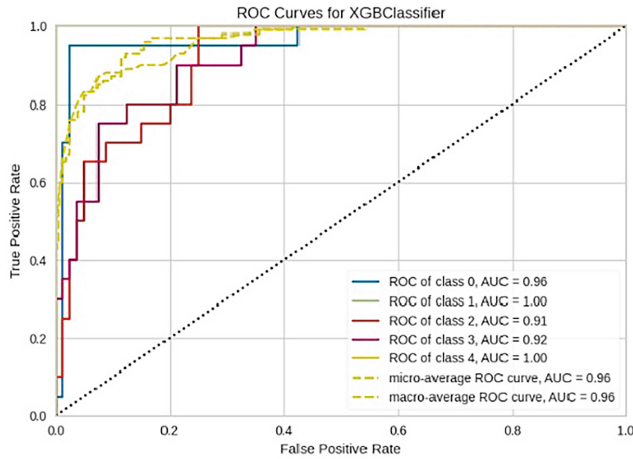


Fig. 6. ROC AUC Curve for XGBoost.

As shown in the confusion matrix of the LinearSVM model in Fig. 3, the model correctly classified all twenty test images, achieving flawless classification accuracy for class 1. However, it showed the weakest performance for class 3, correctly classifying only three test images. These results are consistent with the LinearSVM ROC AUC values presented in Fig. 8. The LinearSVM model achieved a perfect match for class 0, with a ROC AUC value of 1.0. On the

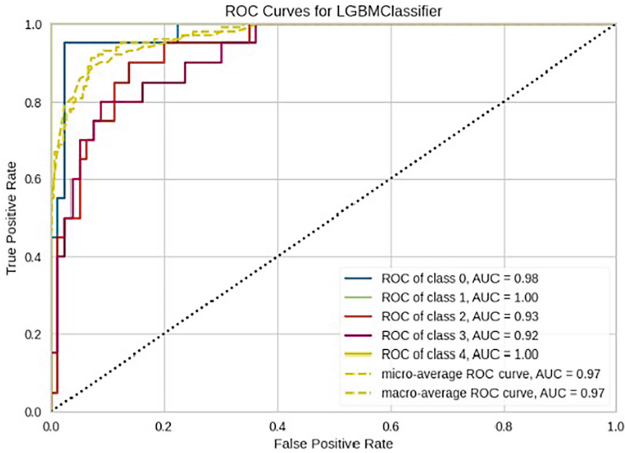


Fig. 7. ROC AUC Curve for LGBM.

other hand, it obtained the lowest ROC AUC value for class 3, which recorded 0.85. The confusion matrix of a fully connected ResNet50, which is utilized in an end-to-end manner including feature extraction and the final classification task. The model demonstrates its greatest proficiency in the classification task for class 1, as it only misclassified a single test image as class 0 out of a total of twenty test images. This is then subsequently followed by classes 3, 0, and 4 which correctly received 15, 13, and 13 correct test images. Twelve test images were incorrectly classified as class 1 and class 3, resulting in the poorest score for class 2. ResNet50 demonstrated the highest level of prediction power for test images in class 1, as evidenced by its ROC AUC value of 0.97 (Fig. 9). A Receiver Operating Characteristic Area Under the Curve (ROC AUC) value of 0.93 suggests that the model has an equal ability to distinguish between test images from class 0 and class 2. Following this, class 4 achieved a ROC AUC value of 0.89, while class 3 obtained a ROC AUC value of 0.87.

Table 9. Quantitative comparison of models

Model	Test Images	Correctly Classified	Misclassified	RMSE	MAE	Accuracy
LinearSVM	100	69	31	1.023	0.5	0.69
GNB	100	64	36	1.096	0.60	0.64
RF	100	78	22	1.023	0.38	0.78
ResNet50	100	68	32	1.127	0.53	0.68
XGBoost	100	78	22	0.878	0.35	0.78
LGBM	100	81	19	0.768	0.33	0.81

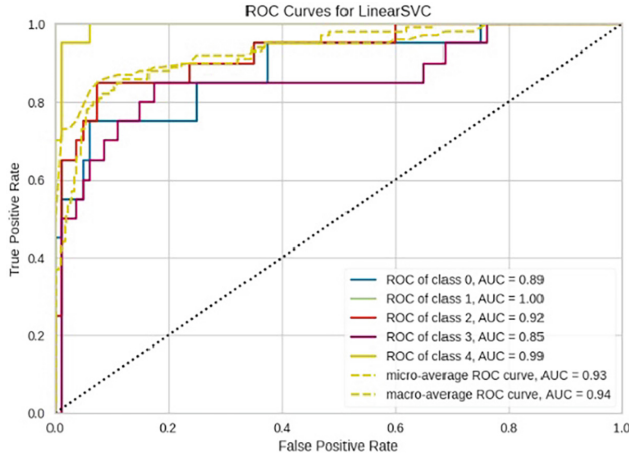


Fig. 8. ROC AUC Curve for LinearSVM.

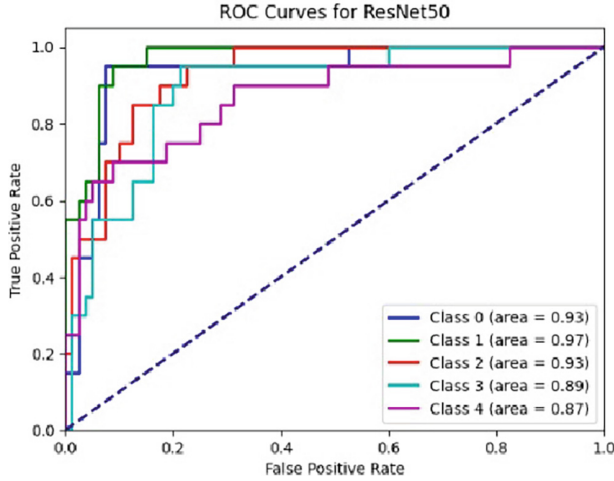


Fig. 9. ROC AUC Curve for ResNet50.

Table 9 shows that the hybrid model using LGBM outperformed others in terms of accuracy, as well as in various evaluation metrics from the confusion matrix, including the counts of correctly and incorrectly classified images, RMSE, and MAE. The results further show that the LGBM-based model demonstrates the highest predictive accuracy, correctly classifying 81 out of 100 test images. It is followed by XGBoost and RF, which both successfully classified 78 test images. Additionally, the LGBM-based model produced the lowest Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) values, at 0.33

and 0.768, respectively. This suggests that, compared to the other models, its predictions are significantly closer to the true values.

Table 10 illustrates that the deep learning model based on LGBM outperformed several other models, such as Mandhala et al. [24] for classifying real-world scenes into categories like coast, forest, highway, and streets; Xia et al. [2] for classifying natural images; Rout et al. and Zheng et al. [25] for evaluating aerial scene classification; Song et al. [26] for indoor scene classification; and Song et al. [27] for RGB-D scene recognition. However, the study by Los et al. [28], which applied LGBM for tree species classification using SENTINEL-2 data, achieved a higher accuracy of 0.94. This difference may be due to their use of a single satellite image and a class category drawn from the same domain.

Table 10. Accuracies obtained from other models

Model	Accuracy
Deep learning model [2]	0.64
SVM model [24]	0.72
Multi-Kernel Learning model [26]	0.61
deep learning model [25]	0.66
Multi-modal combination method [27]	0.67
Recurrent Neural Network (RNN) model [29]	0.67
LGBM model [28]	0.94

6 Conclusion

This study uses a hybrid approach that combines deep convolutional neural networks with traditional machine learning classifiers to categorize images of natural scenes into five distinct classes: building, forest, glacier, mountain, and sea. In order for machine learning classifiers to categorize a test image into one of the five classes, a set of features was extracted using the ResNet50, which was adopted through transfer learning technique. The models' performance was evaluated using metrics such as accuracy and RMSE. With the lowest RMSE of 0.768 and an accuracy of 81%, the LGBM-based model showed the best performance. In the field of forestry, this model can be utilized to categorize trees in accordance with the species to which they belong, as well as to classify vegetation into a variety of various types and categories. With the aim of increasing classification accuracy, the study recommends the adoption of ontologies in the classifier, so that the classification process is informed by ontologies of a specific domain. Explainable AI (XAI) methods like SHAP or LIME may be used in future research to shed light on how the ResNet50 and machine learning classifiers make decisions. Furthermore, combining XAI with attention mechanisms may improve interpretability by highlighting the areas of the image that are most important for classification.

References

1. Chen, C.H.: Handbook of Pattern Recognition and Computer Vision. World Scientific (2015)
2. Rout, A.R., Bagal, S.B.: Natural scene classification using deep learning. In: 2017 International Conference on Computing, Communication, Control and Automation (ICCUBE), pp. 1–5. IEEE (2017)
3. Kumar, V.V., Reddy, K.S., Krishna, V.V.: Face recognition using prominent LBP model. *Int. J. Appl. Eng. Res.* **10**(2), 4373–4384 (2015)
4. Kumari, Y.S., Kumar, V.V., Satyanarayana, C.: Texture classification using complete textron matrix. *Int. J. Image Graph. Signal Process.* **9**(10), 60 (2017)
5. Muhammad, K., Ahmad, J., Baik, S.W.: Early fire detection using convolutional neural networks during surveillance for effective disaster management. *Neurocomputing* **288**, 30–42 (2018)
6. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. *Neural Comput.* **29**(9), 2352–2449 (2017)
7. Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., Lopez, A.: A comprehensive survey on support vector machine classification: applications, challenges and trends. *Neurocomputing* **408**, 189–215 (2020)
8. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
9. Chen, H.-Y., Su, C.-Y.: An enhanced hybrid mobilenet. In: 2018 9th International Conference on Awareness Science and Technology (iCAST), pp. 308–312. IEEE (2018)
10. Kishore, B., Vijaya Kumar, V., Sasi Kiran, J.: Classification of natural images using machine learning classifiers on graph-based approaches. In: Chillarige, R.R., Distefano, S., Rawat, S.S. (eds.) ICACII 2019. LNNS, vol. 119, pp. 319–330. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-3338-9_37
11. Wang, P., Fan, E., Wang, P.: Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. *Pattern Recogn. Lett.* **141**, 61–67 (2021)
12. Li, Y., Dixit, M., Vasconcelos, N.: Deep scene image classification with the mfaivnet. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5746–5754 (2017)
13. Wang, L., Sun, Y.: Image classification using convolutional neural network with wavelet domain inputs. *IET Image Process.* (2022)
14. Breiman, L.: Random forests. *Mach. Learn.* **45**, 5–32 (2001)
15. Ghosh, A., Fassnacht, F.E., Joshi, P.K., Koch, B.: A framework for mapping tree species combining hyperspectral and lidar data: role of selected classifiers and sensor across three spatial scales. *Int. J. Appl. Earth Obs. Geoinf.* **26**, 49–63 (2014)
16. Guan, H., Li, J., Chapman, M., Deng, F., Ji, Z., Yang, X.: Integration of orthoimagery and lidar data for object-based urban thematic mapping using random forests. *Int. J. Remote Sens.* **34**(14), 5166–5186 (2013)
17. Lawrence, R.L., Wood, S.D., Sheley, R.L.: Mapping invasive plants using hyperspectral imagery and breiman cutler classifications (randomforest). *Remote Sens. Environ.* **100**(3), 356–362 (2006)
18. Belgiu, M., Drăguț, L.: Random forest in remote sensing: a review of applications and future directions. *ISPRS J. Photogramm. Remote. Sens.* **114**, 24–31 (2016)
19. Chang, Y.-C., Chang, K.-H., Wu, G.-J.: Application of extreme gradient boosting trees in the construction of credit risk assessment models for financial institutions. *Appl. Soft Comput.* **73**, 914–920 (2018)

20. Kwenda, C., Gwetu, M.V., Fonou-Dombeu, J.V.: Forest image classification based on deep learning and xgboost algorithm. In: International Conference on Computational Science, pp. 217–229. Springer (2023)
21. Ke, G., et al.: Lightgbm: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
22. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
23. Punnet, B.: Intel Image Classification image scene classification of multiclass (1999)
24. Mandhala, V.N., Sujatha, V., Devi, B.R.: Scene classification using support vector machines. In: 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies, pp. 1807–1810. IEEE (2014)
25. Xia, G.-S., et al.: AID: a benchmark data set for performance evaluation of aerial scene classification. *IEEE Trans. Geosci. Remote Sens.* **55**(7), 3965–3981 (2017)
26. Zheng, Y., Gao, X.: Indoor scene recognition via multi-task metric multi-kernel learning from RGB-D images. *Multimedia Tools Appl.* **76**, 4427–4443 (2017)
27. Song, X., Jiang, S., Herranz, L.: Combining models from multiple sources for RGB-D scene recognition. In: IJCAI, pp. 4523–4529 (2017)
28. Łoś, H., et al.: Evaluation of Xgboost and Lgbm performance in tree species classification with sentinel-2 data. In: 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, pp. 5803–5806. IEEE (2021)
29. Song, X., Jiang, S., Wang, B., Chen, C., Chen, G.: Image representations with spatial object-to-object relations for RGB-D scene recognition. *IEEE Trans. Image Process.* **29**, 525–537 (2019)



Ensemble CNNs for Solar Flare Image Classification

Mangaliso Mngomezulu¹ and Mandlenkosi Gwetu²

¹ School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal,
Pietermaritzburg 3021, South Africa
219030015@stu.ukzn.ac.za

² Department of Industrial Engineering, Stellenbosch University, Stellenbosch 7600, South
Africa
mgwetum@sun.ac.za

Abstract. Solar events originate from the sun and normally manifest within its solar system. Solar flares release lethal radio-active energy into space and have been theorized to trigger coronal mass ejection (CME) with lethal terrestrial impacts. The National Aeronautics and Space Administration (NASA)'s Solar Dynamics Observatory(SDO) has captured high-quality images of solar flares that can be used for solar flare studies. Expert physicists struggle with the classification of the images due to their large data size. Convolutional Neural Networks (CNNs) have proven effective with large datasets. This study improves the classification accuracy of solar flares observed at wavelength $1,600\text{ \AA}$ from 94% to 100% by utilizing an ensemble of diverse CNN models that are deliberately set up to be more effective than the best base learners. Data from the Atmospheric Imaging Assembly(AIA) containing three classes of solar flares: limb, two ribbon, and compact, was used. A 'background' class was included for images with no solar flare. The CNN models used are AlexNet, LeNet5, XceptionNet, NASNetLarge, and ResNet50. The best CNNs were AlexNet, LeNet5, and NASNetLarge with f1-scores and accuracies of 99%. The Soft and Hard voting ensembles outperformed the best individual CNNs with an f1-score of 100% and an accuracy of 100%. The conclusion which supports previous literature from other domains, is that practically demonstrable base learner diversity improves an ensemble's performance.

Keywords: Solar Flare · Solar Dynamics Observatory(SDO) · Convolutional Neural Network (CNN) · Base Learner · Ensemble CNN · Diversity

1 Introduction

The solar disk (sun's surface) has magnetic forces as a result of electrical charges from its contained materials [13]. Internal disturbances in the sun release some of the

Supported by the University Of KwaZulu-Natal(UKZN) and South African Radio Astronomy Observatory(SARAO).

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2025

Published by Springer Nature Switzerland AG 2025. All Rights Reserved

T. M. N. Ngatched et al. (Eds.): PAAISS 2024, LNICST 632, pp. 55–69, 2025.

https://doi.org/10.1007/978-3-031-94439-0_4

solar energy to the solar disk and the surrounding environment. Solar events in general terms are various forms of disturbances. Solar flares, coronal mass ejections, high-speed solar winds, sunspots, and prominence are examples of solar events. Solar flares trigger CMEs, which interrupt the Global Positioning System (GPS) in space [26] by increased radiation levels. On the 6th September 2017, coronal mass ejections (CMEs) and a sunspot created a powerful short wave resulting in a radio blackout in the Atlantic Ocean, Africa, and Europe. Communication systems are prone to disturbances triggered by solar events [26]. Astronauts can easily be killed by the radiation levels. Radiation caused by solar flares can directly harm the Earth. Solar flare occurrence rates increase every decade [17]. Early detection of solar flares can aid with mitigation measures. On the 11th February 2010, the Solar Dynamic Observatory (SDO) rocket was launched to collect data on the sun. The data used in this study was obtained in [12], it was published by the Applied Mathematics department of the University of St Andrews (England) in 2020, under the Creative Commons (CC) licensing. The original source of the data is the Atmospheric Imaging Assembly (AIA) which is a tool of the aforementioned SDO.

This study aims to use effective CNNs and lightweight CNN ensembles to classify solar flares observed at wavelength $1,600\text{\AA}$. The CNNs used in the study are pre-existing and well known. The contribution of this study is the employment of well justified strategies and criteria in the construction of the ensemble CNNs that perform better than the best base learners (the pre-existing CNNs). Full descriptions of the solar flare classes and dataset can be found in [12]. The remainder of the paper is as follows, Sect. 2 presents related previous works. Section 3 describes the methods used. Section 4 discusses results obtained & Sect. 5 concludes the paper.

2 Related Works

The first study for visual recognition of solar flare ribbons (captured at wavelength $1,600\text{\AA}$) using CNNs was conducted in [12]. The study focused on classifying solar flare ribbons into one of four classes namely limb flares, compact ribbon flares, two-ribbon flares, and a control class named quiet sun for a solar disk region with no solar flare going on. The data used was obtained from the SDO's Atmospheric Imaging Assembly (AIA), limited to only $\geq C$ class solar flares. The results showed a maximum of 94% classification accuracy. The subjective visual complexity of the observations was relatively low for observations at wavelength $1,600\text{\AA}$. It can be argued that the architecture of the CNN model used by the researchers was either too small, too large, or poorly optimized in training. Another study [9] demonstrated how some CNN architectures can perform in a more complex dataset where the task was to visually recognize various solar events. In consideration of those facts, this study shows that the use of sufficient architectures can allow for improved performance. Also, it shows that more complex model architectures can lower the performance. The study in [9] recommended the use of ensembles for better success of CNNs. Simply combining models to form an ensemble is not a reliable method for getting the best out of an ensemble architecture. An intensive review by [16] discussed the various factors to be considered when setting up the ensemble. One of the key things is ensuring diversity in the base learners

of the ensemble. This study sets up diverse base learners whose diversity is mathematically quantified and produces an ensemble that improves the classification results of the wavelength 1,600 Å observations to 100%. While individual model architecture selection can improve performance, where one model is not efficient an ensemble model can be a more effective solution.

Ensemble CNNs have previously been used for sunspot (solar event) classification by [21]. The data used is a subset of the SDO's Helioseismic and Magnetic Imager (HMI). In the study, multiple CNN models were constructed and ten of those were combined using XGBoost to form the ensemble model. The ensemble outperformed the individual CNNs. The ensemble yielded a superior average of 92% accuracy with its highest f1 score being 0.95. Base learner diversity was credited for the success of the ensemble. This study uses multiple means to induce base learner diversity.

The relationship between the number of classes in the data and the number of models is vital. Let $N_c \in \mathbb{Z}$ be the number of classes and $N_m \in \mathbb{Z}$ be the number of models. As done in this study, it helps to use $N_m > N_c$ just in case models make different predictions. Also as done in this study, it helps to set N_m such that $N_m \% 2 \neq 0$ and N_m is odd, in the case of voting. This helps avoid ties in the votes. In this study, $N_m = 5$ and $N_c = 4$.

The relationship between an ensemble's base learners is crucial for an ensemble's success. According to an ancient French mathematician Jean Nicolas de Cariat, if $V = \{v_1, v_2, \dots, v_n\}$ is a set of n voters, and the probability $p(v_i)$ that each voter chooses the right option in given options is above 0.5, then combining these voters yields better accuracy. It is therefore critical to ensure that base learners each yield at least 50% success on the data. In this study, all the chosen individual CNNs have accuracies significantly greater than 0.5, with the lowest being 92%. With homogeneous base learners, the bagging and boosting approaches are common [6]. Some of the less computationally expensive ensemble methods include weighted averaging, unweighted averaging, and unweighted majority voting [5].

The diversity of the base learners is highly regarded as the most decisive factor of an ensemble's success [10, 16]. There is no clear definition of diversity [10]. The strengths of base learners on distinct parts of the data help the ensemble perform well [24]. Ensembles use the philosophy of "crowds are wiser than one", not all crowds are wiser though [16]. The ensemble will more likely be successful if base learners' decisions are independent to avoid biases [16]. The combination of each base learner's search spaces helps the ensemble have an even bigger solution search space.

3 Methods

3.1 Data Pre-processing

The data used was adopted from [12] stored as pickle objects. The data was first deserialized, went through image standardization, then shuffling, and finally batching in readiness to be fed to the CNNs. Image sizes of 250 x 250 pixels were used. There were 966 images in the dataset in total. Data was partitioned into 70%, 15%, and 15% for training, testing, and validation respectively. Each image has its class. Each class was equi-proportionally represented in the data partitions.

3.2 Similarities Between The Used Individual CNNs

All the individual CNNs use the softmax activation function in their final layer. The softmax is defined in Eq. 1.

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}} \quad (1)$$

where x_i is the input vector, n is the number of classes and e is the standard Euler's number derived function [19]. The softmax computes the probability that the image belongs to a certain class, in this case, one of the four classes. The softmax was chosen since the data that needs to be classified is multinomial and it deals with multinomial probability distributions well. With exception to the LeNet5 model, the other models used in this study were configured to make use of the categorical cross-entropy loss function, defined in Eq. 2.

$$CE(x_i, y_j) = \sum_{i=1}^C x_i \log(y_j) \quad (2)$$

where y_j is the j^{th} value of the mode's possible outputs and x_i being the actual value expected [7]. The categorical cross entropy was chosen because the problem at hand is a multi-class classification problem. The hidden layers used the Rectified Linear Unit (ReLU) activation function defined in Eq. 3.

$$ReLU(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases} \quad (3)$$

where x is the numerical input. This ReLU's strength is not being affected by the vanishing gradients issue. The ReLU was chosen for its positive influence on the model's training speed in contrast to its well-known alternatives. The final layer has four neurons for each CNN model, classifying an image into one of the four classes in the data. The models were configured with early stopping to stop training if the validation loss does not improve after some epochs, this was done to save time. The main means for reducing (optimizing) training time are: using the ReLU, using the EarlyStopping configuration, model optimizers, and the GPU configuration for model training.

3.3 Differences Between the Used Individual CNNs

Differences in CNNs were deliberate means to increase diversity. The use of different optimizers on CNNs increases their diversity [22]. In this study, five different CNN architectures were each configured with a different optimizer. Optimizers were chosen based on their competence as seen in other studies. According to [6], heterogeneous base learners help ensembles to be successful. The AlexNet model used the Stochastic Gradient Descent, ResNet50 used the Adadelata, NASNetLarge used the Adagrad, XceptionNet used the RMSprop and LeNet5 used the Adam optimizer. Table 1 shows how the models differ in their configurations. The learning rate for each model was selected

because it was producing better results than alternatives during the experiments. The optimizers were chosen for performance maximizing and variety amongst the CNNs. Each model was shown to learn best (or faster) when using a certain batch size. The selection of batch sizes also varied amongst most of the models except the AlexNet CNN and the XceptionNet CNN.

Larger models like the ResNet50 required larger batches for efficient training. The number of epochs used was largely dependent on the size of the model, batch size, and the optimizer. This resulted in varied epoch configurations for the models for efficient training. The column named 'Patience Epochs' indicates the number of training epochs for which the model was configured to use as a stopping criterion if the validation performance does not improve for those specific epochs consecutively in training. For example, the 5 for AlexNet means that if the validation loss does not decrease in 5 consecutive epochs, then the model should stop training. This was done to prevent over-fitting. The choice for patience epochs was subjective but based on experimental observations. This number can vary depending on the complexity of the model and the data.

Table 1. Configurations For CNN Models

Model	Learning Rate	Optimizer	Batch Size	Epochs	Patience Epochs	Momentum
AlexNet	0.002	SGD	32	25	5	0.5
XceptionNet	0.0011	RMSProp	32	60	11	0.001
LeNet5	0.015	Adam	16	18	8	—
NASNetLarge	0.0017	Adagrad	24	65	10	—
ResNet50	0.0018	Adadelta	64	100	6	—

The following bullet points each provide justifications of relevance of the methods used. Each model is justified for its success in a relevant application, even if it was in a different context. It is important to note that the bullet points are not meant to provide a comprehensive review of the relevant studies but a brief justification of relevance. Tables 2, 3, 4, 5, and 6 each shows the architectural details of the CNN models used. In cases where a base model was used, it was taken from TensorFlow's Python framework developed by Google Brain [1]. Models whose bases were taken from TensorFlow were imported without pre-trained parameters. The additional layers were customized in the experiment for this study to best suit the models for the task at hand (e.g., the obvious final layer with 4 classes). Other layers were added to create variety while maximizing performance and creating room for exploring how larger models will deal with the problem, e.g., the ResNet50 model. This was done because, in Chap. 1, it was hypothesized that the architecture of models has an impact on performance. The use of the dropout was done to combat over-fitting.

- **AlexNet CNN** - AlexNet was introduced by [8] in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) where it was the top-performing model. So the model on its own has a good reputation. The properties of medical images are very

similar to those of solar flare ribbons [2]. A study [18] shows how the AlexNet model was successful in the visual recognition of diabetic retinopathy images. It obtained at least 95.6% accuracy in various stages of diabetes. The images of the retina used in that study are very relatable to the problem at hand. Its original architecture can be found in [25], which shows that it is a simple model that is easy to train. This model can do well in medical imaging which led to the justification of its relevance in this context based on the aforementioned facts.

Table 2. The Fine-tuned AlexNet CNN Architecture

Layer Type	Parameters	Output Shape
Convolutional	Filters: 128, Kernel: (11, 11), Strides: (4, 4)	(62, 62, 128)
Batch Normalization	–	(62, 62, 128)
Max Pooling	Pool Size: (2, 2)	(31, 31, 128)
Convolutional	Filters: 256, Kernel: (5, 5), Strides: (1, 1)	(31, 31, 256)
Batch Normalization	–	(31, 31, 256)
Max Pooling	Pool Size: (3, 3)	(10, 10, 256)
Convolutional	Filters: 256, Kernel: (3, 3), Strides: (1, 1)	(10, 10, 256)
Batch Normalization	–	(10, 10, 256)
Convolutional	Filters: 256, Kernel: (1, 1), Strides: (1, 1)	(10, 10, 256)
Batch Normalization	–	(10, 10, 256)
Convolutional	Filters: 256, Kernel: (1, 1), Strides: (1, 1)	(10, 10, 256)
Batch Normalization	–	(10, 10, 256)
Max Pooling	Pool Size: (2, 2)	(5, 5, 256)
Flatten	–	6400
Dense (Fully Connected)	Neurons: 1024, Activation: ReLU	1024
Dropout	Rate: 0.5	1024
Dense (Fully Connected)	Neurons: 1024, Activation: ReLU	1024
Dropout	Rate: 0.5	1024
Dense (Output)	Neurons: 4, Activation: Softmax	4

- **XceptionNet CNN** - In a COVID-19-based study [20], A modified version of XceptionNet demonstrated success in image recognition when compared to some of the most recent models. As mentioned earlier, medical images are similar to solar events' images, this makes the use of XceptionNet more reliable. This study however uses the standard XceptionNet model with modifications as presented in Table 3.

Table 3. The Fine-tuned XceptionNet-based Neural Network Architecture

Layer Type	Parameters	Output Shape
Xception Base	Input Shape: (250, 250, 1), Pooling: Max	(None, 2048)
Dense (Fully Connected)	Neurons: 512, Activation: ReLU	512
Dense (Fully Connected)	Neurons: 64, Activation: ReLU	64
Dense (Output)	Neurons: 4, Activation: Softmax	4

- **LeNet5 CNN** - The LeNet CNN was first introduced in [11]. Named after Yann LeCun, the model has been applied in the recognition of complex graphical figures like handwritten digits [11]. A later variation named LeNet5 was applied in solar event detection [3], where it was successful. The model has also been successfully applied in predicting the arrival time of coronal mass ejections (solar events) [23]. The slightly tuned version used in this study is shown in Table 4.

Table 4. LeNet-5 Inspired Neural Network Architecture

Layer Type	Parameters	Output Shape
Convolutional	Filters: 32, Kernel: (5, 5), Input Shape: (250, 250, 1)	(250, 250, 32)
Max Pooling	Strides: 2	(125, 125, 32)
Convolutional	Filters: 48, Kernel: (5, 5) Activation: ReLU	(121, 121, 48)
Max Pooling	Strides: 2	(60, 60, 48)
Flatten	–	172800
Dense (Fully Connected)	Neurons: 256, Activation: ReLU	256
Dense (Fully Connected)	Neurons: 84, Activation: ReLU	84
Dense (Output)	Neurons: 4, Activation: Softmax	4

- **NASNetLarge CNN** - In another COVID-19-related study [14] using X-ray images, the NASNetLarge CNN model was among other well-performing models in a recognition task meant to aid an automated diagnosis of COVID-19-related infections. X-ray images fall under the category of medical images. Other recent works also applied this model to medical images [27]. The fine-tuned version of this model used in this study is provided in Table 5.

Table 5. The Fine-tuned NASNetLarge-Based Neural Network Architecture

Layer Type	Parameters	Output Shape
NASNetLarge Base	Input Shape: (250, 250, 1),	(None, None, 4032)
Dropout	Rate: 0.5	(None, None, 4032)
Flatten	–	4032
Batch Normalization	–	4032
Dense (Fully Connected)	Neurons: 32, Kernel Initializer: He Uniform	32
Batch Normalization	–	32
Activation (ReLU)	–	32
Dropout	Rate: 0.5	32
Dense (Fully Connected)	Neurons: 32, Kernel Initializer: He Uniform	32
Batch Normalization	–	32
Activation (ReLU)	–	32
Dense (Output)	Neurons: 4, Activation: Softmax	4

- **ResNet50 CNN** - In the task of brain tumor detection, a study [4] showed that the ResNet50 model was a good candidate for the recognition/classification of images with MRI scans. MRI scans are medical images and these are similar to solar events' images, thus making the ResNet50 model a good candidate base learner. The model has been slightly fine-tuned for use in this study and the resulting architecture is summarized in Table 6.

Table 6. The Fine-tuned ResNet50-Based Neural Network Architecture

Layer Type	Parameters	Output Shape
ResNet50 Base	Input Shape: (250, 250, 1)	(None, None, 2048)
Dropout	Rate: 0.5	(None, None, 2048)
Flatten	–	2048
Batch Normalization	–	2048
Dense (Fully Connected)	Neurons: 2048, Kernel Initializer: He Uniform	2048
Batch Normalization	–	2048
Activation (ReLU)	–	2048
Dropout	Rate: 0.5	2048
Dense (Fully Connected)	Neurons: 1024, Kernel Initializer: He Uniform	1024
Batch Normalization	–	1024
Activation (ReLU)	–	1024
Dropout	Rate: 0.5	1024
Dense (Output)	Neurons: 4, Activation: Softmax	4

3.4 Measuring Diversity of Individual CNNs

Table 7 shows some tabulated metrics of the diversity measure on some of the models. Previous utilization of the diversity measure can be found in [10]. Let P_{m_1, m_2} be a prediction of models m_1 and m_2 on some test data. Assume that 0 and 1 mean wrong and correct classification, respectively, such that $P_{0,1}$ is a set of cases were m_1 was wrong and m_2 was correct. Using a similar analogy, the diversity measure for m_1 and m_2 is:

$$D_{m_1, m_2} = \frac{|P_{0,1}| + |P_{1,0}|}{|P_{0,1}| + |P_{1,0}| + |P_{0,0}| + |P_{1,1}|} \quad (4)$$

where D_{m_1, m_2} measures how often the models m_1 and m_2 disagree when they are tested. For the sake of presentation. Table 7 shows the disagreement of the models. The weakness of this metric is that it does not give clues about whether models were both wrong in their disagreements or not.

A better alternative method to measure diversity is the Q-statistic as discussed in [10]. Its result can be used to deduce more information about a pair of classifiers than

Table 7. The disagreement measure for some of the CNNs

	AlexNet	XceptionNet	LeNet5	NASNetLarge	ResNet50
AlexNet	—	2.055×10^{-2}	6.85×10^{-3}	1.370×10^{-2}	8.23×10^{-2}
XceptionNet	—	—	1.370×10^{-2}	2.050×10^{-3}	8.90×10^{-2}
LeNet5	—	—	—	6.85×10^{-3}	8.90×10^{-2}
NASNetLarge	—	—	—	—	8.21×10^{-2}
ResNet50	—	—	—	—	—

with the disagreement measure. Using a similar notation with the disagreement measure, the Q-statistic for 2 classifier models m_a and m_b is computed as:

$$Q_{a,b} = \frac{(|P_{1,1}| * |P_{0,0}|) - (|P_{0,1}| * |P_{1,0}|)}{(|P_{1,1}| * |P_{0,0}|) + (|P_{0,1}| * |P_{1,0}|)} \quad (5)$$

For any pair of classifiers, m_a and m_b , $Q_{a,b} > 0$ implies that m_a and m_b have a tendency of making correct predictions. $Q_{a,b} < 0$ implies that m_a and m_b have a tendency to both make wrong predictions. $\forall Q_{a,b}$, $Q_{a,b} \in [-1, 1]$, $Q_{a,b} \in \mathbb{R}$. The computation of the Q-statistic show that both the AlexNet and XceptionNet have a value of -1 against the NASNetLarge and also against each other. Both the AlexNet and XceptionNet have a value of 0 against the LetNet5. The AlexNet and NASNetLarge both have a value of 1 against the ResNet50 model. The XceptionNet has a value of 0.563 against the ResNet50 model and lastly the LeNet5 with a value of 0 against the ResNet50. There are more non-negative values in the Q-statistic values for pairs of models, which is a good indication of the models' correct classifications. The negative values indicate room for the models to complement each other's weaknesses in the ensemble model if they can.

3.5 Computing Platform Specifications

The individual models were trained sequentially. The Google Collab environment was utilized for the experiments. The Graphics Processing Unit (GPU) configuration was used to save time. The device details are Intel (R) Xeon (R) CPU @2.00 GHz, Random Access Memory (RAM): 13297228 Kilobytes, CPU cores: 2 threads per core and CPU, Programming Language used were Python3 (3.7.15) and C++.

3.6 Training of Models

The individual models were trained sequentially. The Google Collab environment was utilized for this experiment. The Graphics Processing Unit (GPU) configuration was used to save time. The device details are Intel (R) Xeon(R) CPU @2.00 GHz, Random Access Memory (RAM): 13297228 Kilobytes, CPU cores: 2 threads per core and CPU, Programming Language: Python3(3.7.15).

The shapes and smoothness of the graphs in Fig. 1 show the variations among models. To be presentable, the training and validation losses were squashed from the range

$[0, \infty]$ to $(0, 1)$ using the sigmoid (normally used as an activation function), defined as $\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$. This comes from the fact that $\forall x, x \in \mathbb{R}, \sigma(x) \in (0, 1)$. What is more important to observe is the trend/shape of the graphs over time than their original values, for this study.

The AlexNet & NASNetLarge models demonstrated a good discrimination ability at around 13 and 40 epochs respectively and going forward. This is evident in the AUC curves sampled in Fig. 1. The XceptionNet CNN seemed to need more time to have stable AUC and validation losses. The Early Stopping configuration for the models was to stop training if the model go for a certain number of epochs without an improvement in the validation loss. The XceptionNet model in fact stopped early due to the Early Stopping configuration.

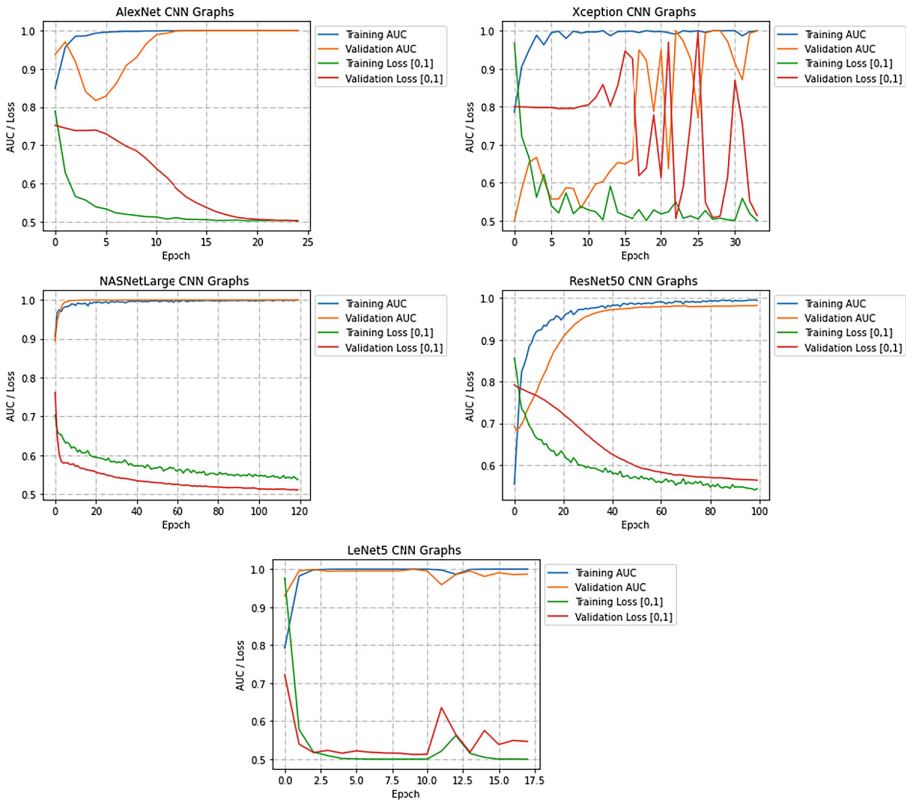


Fig. 1. Sample AUC, Validation (AUC), Training Loss, Validation Loss Curves for CNN Models

The GPUs helped decrease the training time for models. The training time descending order is NASNetLargeNetLarge, XceptionNet, ResNet50, AlexNet, and LeNet5. The AlexNet and LeNet5 CNNs are not only required less time, but they also performed well. In Fig. 1 it can be noted that for the NASNetLarge model, the validation loss curve

is below the training loss curve. This is due to the effect of the sigmoid applied on the loss curve to constrain it in the range $[0, 1]$ for the sake of being presentable.

3.7 Evaluation Metrics

The metrics used to measure the success of the classifiers were namely accuracy, sensitivity, specificity, and f1-score. The equations for these metrics can be found in [9].

4 Results and Discussion

4.1 Confusion Matrices From CNNs

Figure 2 shows that all models on average did well in recognizing the Limb & Two Ribbon (labeled as '2 Ribbon') class. With a common misclassification of the Compact class. The simplest class to recognize is the Background, the confusion matrices show that there were no false positives in that class. As the average accuracy shows, the AlexNet model average had more true positives, with some false positives of the Limb class when the true positives would have been Compact. As evident on the confusion matrices, LeNet5 & XceptionNet misclassified the Compact & Background events, what may be interesting is that for each of those misclassifications, the models had false positives on different classes. In overall, the false alarm rates (FAR) were low, meaning the models performed well and can be usable in a real-world setting.

The key performance metrics: accuracy (Acc), specificity (Spec), sensitivity (Sens), and F1-score, are displayed in Table 8 for individual models.

4.2 Ensemble CNNs and Their Performance

The choice of different CNN architectures was motivated by the aforementioned prerequisite for ensemble success, diversity. In this experiment, it was observed that the error correlation of the models was low. In this study, more than two classifiers are used, making diversity quantification a slightly challenging task as stated by [10]. The Soft-Voting ensemble is mathematically defined as: $\hat{y} = \operatorname{argmax}_j \sum_{i=1}^n w_i p_{i,j}$ where $p_{i,j}$ is the predicted class of the i^{th} classifier for the j^{th} class label. The term $w_i, \forall w_i \in \{w_1, w_2, \dots, w_n\}$ is a weight parameter that is optional, its default value is $\frac{1}{n}$ where the weights are equally shared. In this study, the default weight was used. To define the Hard-Voting/Majority Voting ensemble, let $S = \{m_1, m_2, m_3, \dots, m_n\}$ denote a set of n models. Then if x is one of the test data items and $p(m_i(x))$ is the prediction of model i given on x , the ensemble vote on x 's classification is defined as:

$$\hat{z} = \operatorname{mode}\{p(m_1(x)), p(m_2(x)), p(m_3(x)), \dots, p(m_n(x))\}.$$

Both the Hard Voting and Soft Voting ensemble did very well obtaining a 100% accuracy in the final tests on average. A tabulated summary of previous works can be seen in Table 9. The results from this study agree with [9] on the prediction that ensemble CNNs would do better than individual CNNs.

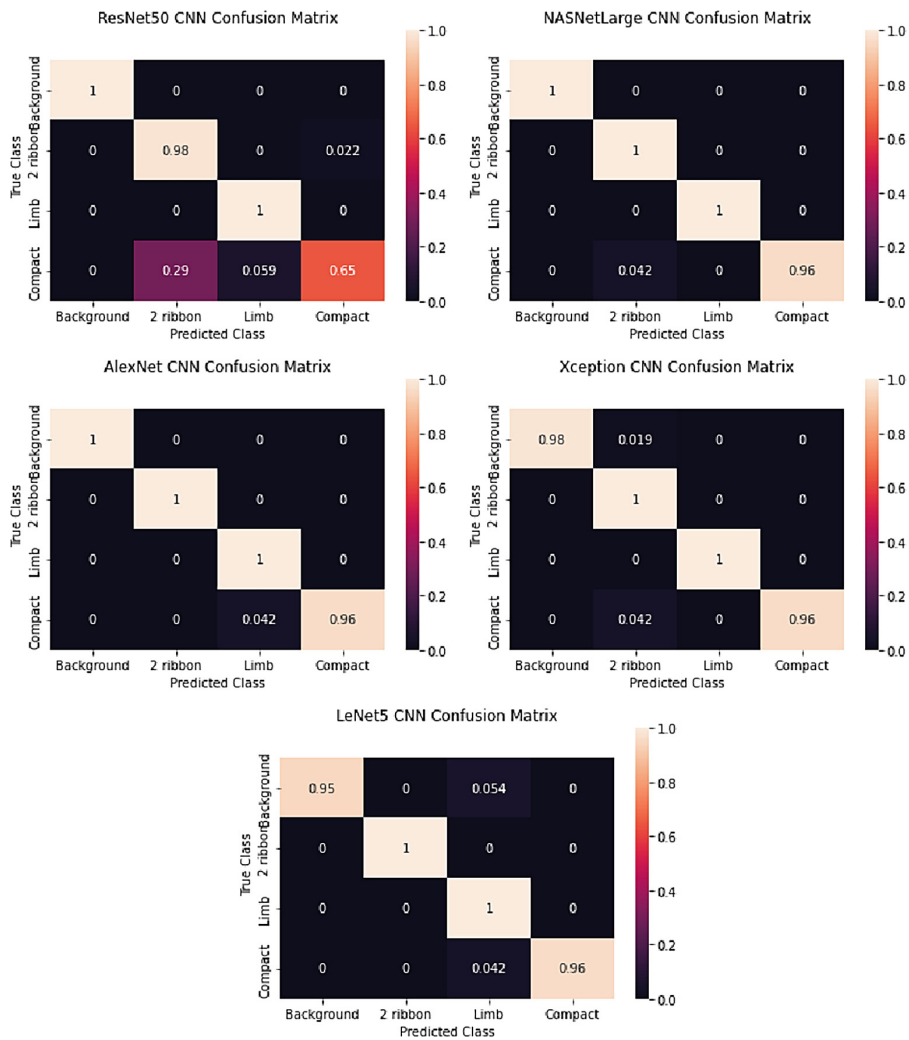


Fig. 2. Sample Confusion Matrices for CNN Models

Table 8. Individual CNN Model Performance Summary

CNN	Acc	Spec	Sens	AUC	F1-score
AlexNet	0.99	0.99	0.98	0.99	0.99
XceptionNet	0.98	0.98	0.96	0.97	0.94
LeNet5	0.99	0.99	0.98	0.99	0.99
NASNetLarge	0.98	0.99	0.98	0.99	0.99
ResNet50	0.92	0.96	0.90	0.99	0.9

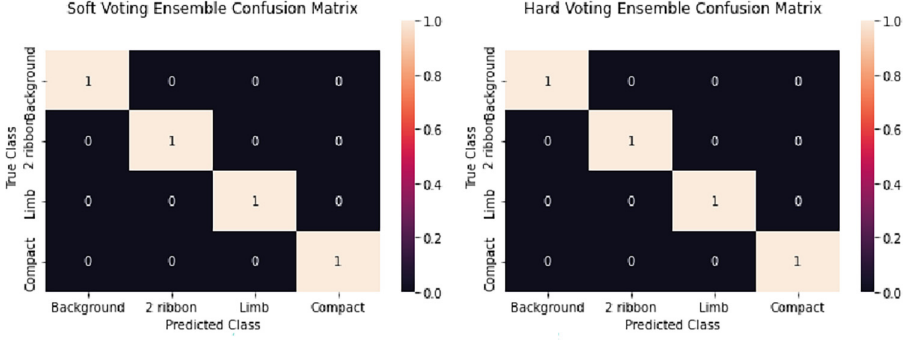


Fig. 3. Confusion Matrices for Soft and Hard Voting Ensemble

Table 9. Related Works & Classification Accuracies

Paper Reference	Background	Compact	Two Ribbon	Limb
Love et al., 2020 (Individual CNN)	95%	88%	97%	92%
This study(Ensemble CNNs)	100%	100%	100%	100%

The following observation was noted when conducting this study. The overall conclusion is that error correlation need not always be low as most studies claim, it can be high at times and have no effect on the ensemble's choices. Let $P_m = \{m_i p(x), m_{i+1} p(x), m_{i+2} p(x), \dots, m_n p(x)\}$ be a set of predictions on input x by n models, where $n \in \mathbb{Z}$, such that $m_i p(x)$ is the i^{th} model's classification prediction on input x . In this experiment, $n = 5$, the five models are namely AlexNet, LeNet5, ResNet50, NASNetLarge, and XceptionNet. Define $W_m(x) \subseteq P_m(x)$ where $W_m(x)$ is the set of wrong classifiers on predicting x 's class. In this study, it was noted that if $|W_m(x)| < |P_m(x) \setminus W_m(x)|$ then the error correlation of models in $W_m(x)$ will have no impact on the ensemble's final correct classification, even if the models in $W_m(x)$ are fully correlated in their predictions. This fact applies to the hard-voting ensembles. For the aforementioned fact to be true in soft voting ensembles, it is required that the sum of probabilities for the correct class in the models in $P_m(x) \setminus W_m(x)$ must be greater than the sum of probabilities of any class in $W_m(x)$.

Initially, the null hypothesis H_o was that there would be no difference between an ensemble and the best individual CNN. A McNemar's statistical significance test [15] was used to test H_o . To avoid division by 0, the prediction count values were increased by 1. This follows after the ensemble had 100% accuracy. The results show that the Chi-square statistic is ≈ 13.482 & p-value $\approx 2.440 \times 10^{-4}$, using $\alpha = 0.05$. The statistical significance implies that H_o is rejected.

In real world operational settings there is a possibility that the ensemble models will not be a silver bullet for accurate solar flare classification. Implementing ensembles requires more compute resources than just one CNN model. On a large scale, trade-offs between operational costs and accuracy may results in the best base learner being the practically feasible candidate. The response time of an ensemble is at best (shortest) the

same as that of the base learner with the longest response time plus the time it takes to consolidate the predictions. This applies if the base learners are tested in parallel. In sequential testing the response time of the ensemble is at least the sum of the reaction times of all its base learners plus the prediction consolidation time. Also, if very unfamiliar data comes at test phase, if the base learners fail to be diverse when they are wrong then the ensemble loses its predictive power.

5 Conclusion

This study improves on results obtained by [12] in the classification of solar flares observed at wavelength $1,600\text{ \AA}$ by a margin of 6% incremental accuracy. Both studies use exactly the same data from the Atmospheric Imaging Assembly (AIA) published by the Applied Mathematics Department of the University of St Andrews (England) in 2020. In this study, the best individual CNN on average accuracy performance was the AlexNet and LeNet5 models. The best CNN models from the experiment, AlexNet 99%, LeNet5 99%, and NASNetLarge 98%. They each had f1-scores of 0.99. The best individual models in this study performed better than those from [12], with an incremental margin of at least 4% higher accuracy. This can to a large part, be attributed to choosing a more suitable architecture. From this study, the accuracy of the best CNNs and the ensemble has a very low margin $\approx 1\%$. Taking advantage of diverse base learners, in this study the hard and soft voting ensembles were used and they both obtained a 100% accuracy, f1-score, etc. The improvement is a result of deliberate means of base-learner diversity as discussed in the methods used in this study. The emphasis on base learner diversity was motivated by a review from [16] and ensembles alone by [9] in the context of solar flare classification. In the case that the best model gets 100% accuracy, it may not be necessary to construct an ensemble as they need more training time (for sequential training of base learners) or more resources (parallel training of base learners).

In future works, a different dataset and homogeneous and heterogeneous ensembles can be explored for the task of solar flare classification.

References

1. Abadi, M.: Tensorflow: learning functions at scale. In: Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming (2016)
2. Banda, J.M., Angryk, R.A., Martens, P.C.: On the surprisingly accurate transfer of image parameters between medical and solar images. In: 2011 18th IEEE International Conference on Image Processing, pp. 3669–3672. IEEE (2011)
3. Chola, C., Benifa, J.B.: Detection and classification of sunspots via deep convolutional neural network. *Global Transit. Proc.* **3**(1), 177–182 (2022)
4. Çinar, A., Yildirim, M.: Detection of tumors on brain MRI images using the hybrid convolutional neural network architecture. *Med. Hypotheses* **139**, 109684 (2020)
5. Fox, K.C.: Classifying solar eruptions (2012). https://www.nasa.gov/mission_pages/sunearth/news/classify-flares.html. Accessed 03 May 2022
6. Gopika, D., Azhagusundari, B.: An analysis on ensemble methods in classification tasks (2014)

7. Koidl, K.: Loss functions in classification tasks. School of Computer Science and Statistic Trinity College, Dublin (2013)
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **25** (2012)
9. Kucuk, A., Banda, J.M., Angryk, R.A.: Solar event classification using deep convolutional neural networks. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) *ICAISC 2017. LNCS (LNAI)*, vol. 10245, pp. 118–130. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59063-9_11
10. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Mach. Learn.* **51**(2), 181 (2003)
11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
12. Love, T., Neukirch, T., Parnell, C.E.: Analyzing aia flare observations using convolutional neural networks. *Front. Astron. Space Sci.* **7**, 34 (2020)
13. National Oceanic and Atmospheric Administration: Solar activity and space weather (2023). <https://www.swpc.noaa.gov/phenomena/solar-activity-and-space-weather>. Accessed 17 Oct 2022
14. Pun, N.S., Agarwal, S.: Automated diagnosis of covid-19 with limited posteroanterior chest x-ray images using fine-tuned deep neural networks. *Appl. Intell.* **51**(5), 2689–2702 (2021)
15. Roggo, Y., Duponchel, L., Huvenne, J.P.: Comparison of supervised pattern recognition methods with McNemar’s statistical test: application to qualitative analysis of sugar beet by near-infrared spectroscopy. *Anal. Chim. Acta* **477**(2), 187–200 (2003)
16. Sagi, O., Rokach, L.: Ensemble learning: a survey. *Wiley Interdisc. Rev. Data Min. Knowl. Disc.* **8**(4), e1249 (2018)
17. Scientific American: Mysteriously powerful particles from solar explosions unveiled in new study (2016). <https://www.scientificamerican.com/article/mysteriously-powerful-particles-from-solar-explosions-unveiled-in-new-study/>. Accessed 01 May 2022
18. Shanthi, T., Sabeenian, R.: Modified alexnet architecture for classification of diabetic retinopathy images. *Comput. Electr. Eng.* **76**, 56–64 (2019)
19. Sharma, S., Sharma, S., Athaiya, A.: Activation functions in neural networks. *Towards Data Sci.* **6**(12), 310–316 (2017)
20. Singh, K.K., Siddhartha, M., Singh, A.: Diagnosis of coronavirus disease (covid-19) from chest x-ray images using modified xceptionnet. *Roman. J. Inf. Sci. Technol.* **23**(657), 91–115 (2020)
21. Tang, R., et al.: Multiple cnn variants and ensemble learning for sunspot group classification by magnetic type. *Astrophys. J. Suppl. Ser.* **257**(2), 38 (2021)
22. Taqi, A.M., Awad, A., Al-Azzo, F., Milanova, M.: The impact of multi-optimizers and data augmentation on tensorflow convolutional neural network performance. In: 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), pp. 140–145. IEEE (2018)
23. Wang, Y., Liu, J., Jiang, Y., Erdélyi, R.: CME arrival time prediction using convolutional neural network. *Astrophys. J.* **881**(1), 15 (2019)
24. Xiao, Y., Wu, J., Lin, Z., Zhao, X.: A deep learning-based multi-model ensemble method for cancer prediction. *Comput. Methods Programs Biomed.* **153**, 1–9 (2018)
25. Yan, L.C., Yoshua, B., Geoffrey, H.: Deep learning. *Nature* **521**(7553), 436–444 (2015)
26. Zanda, M., Brown, G.: A probabilistic perspective on ensemble diversity. Ph.D. thesis, University of Manchester (2010)
27. Zhang, Y.: Lung segmentation with nasnet-large-decoder net. *arXiv preprint arXiv:2303.10315* (2023)



Natural Language Processing in Automatic Grading of Assessments in Higher Education: A Systematic Literature Review

Lizzy Ofusori¹(✉) , Tebogo Bokaba² , and Siyabonga Mhlongo²

¹ Centre for Applied Data Science, University of Johannesburg, Johannesburg, South Africa
lofusori@uj.ac.za

² Department of Applied Information Systems, University of Johannesburg, Johannesburg, South Africa

Abstract. With the increasing digitization of education and large-scale standardized testing, the use of natural language processing (NLP) for the automatic grading of assessments has surged, leading to the widespread availability and adoption of commercial solutions. Automatic grading is not limited to multiple-choice questions, but also includes essays and short answers, highlighting its potential to automate rapid assessment and feedback in higher education. This study explores various NLP techniques used in automatic grading, investigates research outcomes, identifies recommended NLP approaches, and examines future directions for automated assessment systems. A systematic literature review following PRISMA guidelines was conducted using Scopus data, identifying 32 relevant studies from a pool of 279 peer-reviewed articles. Findings indicate that automatic grading techniques vary based on assessment type and available data. In summary, this study provides an updated overview of NLP applications in assessments, enhancing the understanding of current research and suggesting directions for future research.

Keywords: Natural Language Processing · Assessment · Automatic Grading · Artificial Intelligence · Higher Education

1 Introduction

Assessment plays a critical role in the education system by evaluating student performance. Traditionally, evaluations are conducted manually by teachers. However, as the student-to-teacher ratio increases, this manual process becomes more complicated and time-consuming. Additionally, manual evaluation lacks reliability and consistency [1]. For example, Burrows, Gurevych and Stein [2] study highlights the limitations of manual grading, which include inconsistency and the time-consuming and labor-intensive nature of assessing large volumes of student submissions, particularly in contexts like standardized testing or massive open online courses (MOOCs). However, using AI-powered grading systems, such as automated essay scoring (AES), leverages natural language

processing (NLP) to process and grade assessments in seconds, drastically reducing the workload of educators and allowing for faster feedback. Studies have shown that artificial intelligence (AI) models can evaluate hundreds of responses in the time it takes a human grader to assess a few, which enhances the potential for timely feedback, crucial for student learning and improvement [3, 4]. Educational institutions have widely adopted AI across various service delivery methods for students [1]. Among the most prevalent AI techniques for mining student opinions is NLP, which is crucial for interpreting end-user feedback and opinions [5]. NLP is a subfield of AI that focuses on the interaction between computers and humans through natural language. It involves enabling computers to understand, interpret, and respond to human language in a valuable way [1]. According to Rokade, Patil, Rajani, Revandkar and Shedge [5], common applications of NLP include fighting spam, information extraction, summarization, machine translation, and processing and grading answers. In higher education, using NLP in the automatic grading of assessments plays a crucial role in assessing students' learning abilities [6]. Moreover, meaningful formative and summative assessments are essential to active learning in higher education. According to Ramesh and Sanampudi [6], over the past few decades, researchers have been exploring automated essay grading, short answer scoring systems and automated scoring of multiple-choice questions. Similarly, in a recent study, Gao, Merzdorf, Anwar, Hipwell and Srinivasa [7] examined automated grading of text-based open-ended questions. Their study highlights significant progress in NLP and machine learning (ML) techniques, which have enhanced the capabilities of automatic assessment tools. However, the study notes that while automated assessment systems show promise, their accuracy and reliability are not yet on par with human grading. Additionally, the study shows that there are discrepancies, particularly in evaluating creative or complex responses that require deeper understanding and contextual interpretation. Nevertheless, Rokade, Patil, Rajani, Revandkar and Shedge [5] opine that automating grading and assessment can save teachers time, enhance accuracy and consistency, and offer students more comprehensive feedback.

While automated evaluation is widely available for multiple-choice questions, essay questions, and short answers, it is important to explore the various NLP techniques used in the automatic grading of assessments in higher education, as well as the challenges encountered and the research outcomes in automated assessment systems. Accordingly, three research questions (RQs) were raised:

RQ1. What are the various NLP techniques used in the automatic grading of assessments in higher education?

RQ2. What are the research outcomes of automated assessment systems?

RQ3. What are the recommended NLP techniques and future work on automatic grading of assessments?

These RQs aim to advance understanding, inform evidence-based decision-making, and support the broader adoption of NLP-based technologies in education, ultimately contributing to a more efficient, scalable, and responsive assessment ecosystem.

The rest of the paper is arranged as follows: Sect. 2 explains the research methodology employed for the study. Section 3 presents and discusses the findings as they relate to the RQs, thus expanding the frontiers of knowledge on NLP in automatic grading of assessments. Section 4 presents the conclusion and limitations of the study.

2 Methodology

The researchers conducted a systematic literature review (SLR) to examine the various NLP techniques used in the automatic grading of assessments. An SLR employs structured methods to collect, analyze, and interpret secondary data that is directly related to the research question [8]. This process allows for the collection of pertinent evidence on a given topic that meets predefined eligibility criteria, providing answers to the formulated research questions [9]. This study chose an SLR because it provides a structured, in-depth examination of existing research, offering a broad and detailed view of the NLP methods used in automatic grading. Furthermore, it helps identify current trends in NLP techniques applied to grading, such as sentiment analysis and text classification. Hence, this method was chosen to gather and summarize current evidence on NLP, which was then analyzed to inform how the educational sector can leverage its benefits.

2.1 Data Sources and Selection Criteria

This section outlines the data sources used in the study, detailing the database, search terms, and inclusion/exclusion criteria. This study data was obtained from the Scopus database accessed on 13 August 2024. The Scopus database was selected due to its comprehensive coverage, high-quality content and advanced analytical tools compared to other databases such as the Web of Science (WoS) [10]. According to Zupic and Čater [11], a wider coverage is beneficial for mapping out smaller research areas. This extensive coverage is particularly important in the context of NLP and automatic grading of assessments. The search strategy employed was limited to disciplines such as Computer Science, Engineering, Decision Science, Mathematics and Multidisciplinary. Moreover, articles were assessed and screened for eligibility using the following basic search string applied to all searchable fields of the Scopus database: (*“Natural language processing”* OR *“NLP”*) AND (*“Grading”*) AND (*“Assessment*”* OR *“Test*”* OR *“Exam*”*).

This search string aligns with the research objective which was to explore the various NLP techniques used in the automatic grading of assessments in higher education. Quotation marks were used to capture variations of the search terms. Additionally, further criteria were applied to refine the search results, including: (i) only records authored in English; and (ii) all types of publications except abstracts, corrections, data papers, book reviews, and letters. The publishers include Sage, Elsevier, Institute of Electrical and Electronics Engineers (IEEE), Springer Nature, Taylor & Francis and the Multidisciplinary Digital Publishing Institute (MDPI). The study covered the period from 2014 to 2024. Using the Preferred Reporting Items for Systematic Reviews and Meta-Analyses

(PRISMA) approach as a guide, a total of 279 articles were identified through the Scopus database. The PRISMA standard includes identification, screening, eligibility, and included [12]. As presented in Fig. 1, the PRISMA framework consists of a checklist and a flow diagram to help ensure the transparent and complete reporting of research. The 279 articles were further screened for eligibility, and a total of 32 articles were successfully retrieved and included in the final analysis. Table 1 summarises the considerations and choices made in the study.

Table 1. Overview of search procedure

Aspect	Choice
Database	Scopus
Search query	Title-Abstract-Keywords (“Natural language processing” OR “NLP”) AND (“Grading”) AND (“Assessment*” OR “Test*” OR “Exam*”)
Period	2014 to 2024
Date of query	13 August 2024
Language	English
Categories	All types of publications except abstracts, corrections, data papers, book reviews, and letters; and
Number of documents included	32

The 32 records obtained were subsequently classified into three major article types (as depicted in Table 2). The data extracted were synthesized using the core themes identified. Thematic synopsis was crucial in examining the various NLP techniques used in the automatic grading of assessments. In addition, two independent researchers were consulted to extract data from the selected research papers based on the study’s RQs. The data extracted by the two researchers were compared, and their findings were discussed. Disparities in studies can be minimized through mutual agreement among the researchers involved, who review their codes to ensure consistency and relevance [9]. Table 2 presents the distribution of the reviewed research papers, showing the number of papers on NLP in automatic grading of assessments (2014–2024). A significant percentage of articles (71.9%) were published in journals, followed by conference proceedings (25%), and books (3.1%).

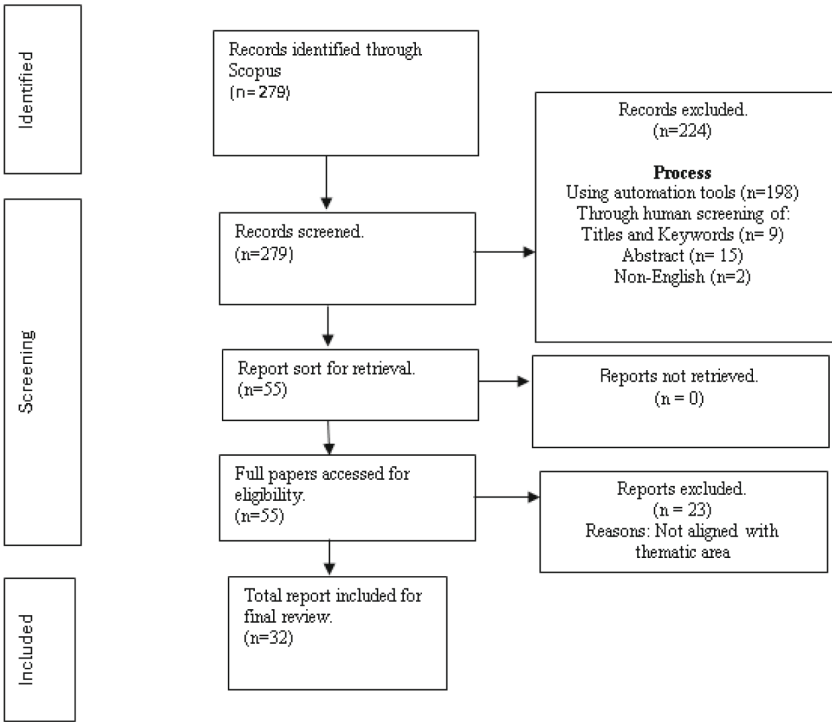


Fig. 1. PRISMA process

Table 2. Summary of the distribution of the reviewed research papers

Reviewed papers	Number of articles	Percentage
Journals	23	71.9
Conference proceedings	8	25
Book	1	3.1

3 Findings and Discussion

This section provides a detailed explanation of the data extracted from the reviewed research papers. The SLR provides a structured and rigorous approach to understanding the various NLP techniques used in the automatic grading of assessments in higher education. Furthermore, it identifies the various research outcomes of automated assessment systems. It investigates the recommended NLP techniques and future work on the automatic grading of assessments. The reviewed papers were mapped as they relate to each research RQ. These are presented in Tables 3, 4 and 5 respectively.

3.1 NLP Techniques Used in the Automatic Grading of Assessments in Higher Education

RQ1 examined the various NLP techniques and datasets used in the automatic grading of assessments in higher education. According to Shaik, Tao, Li, Dann, McDonald, Redmond and Galligan [13], the specific techniques used in automatic grading systems vary depending on the assessment type (essay, multiple choice, short answer), desired level of granularity, and available data. In this subsection, NLP techniques and datasets used by the various authors are discussed. These are summarized in Table 3.

Table 3. NLP techniques used and dataset in automatic grading of assessments

NLP Techniques	Datasets	Algorithm	Authors
Text Classification	W-Pal	Regression-based ML algorithms	[14]
	MOOC	K-Means clustering	[3]
	HMB-EMD-v1	Word2Vec, FastText, Glove, and USE	[4]
	RfU dataset	SVM	[15]
	OS dataset and the Mohler dataset	LLM	[16]
Text Summarization	Compiled descriptive responses in an educational setting	RAKE-ROUGE	[17]
	Not specified	Recursive Neural Network	[18]
Sentiment Annotation	Twitter sample dataset	LSTM, Naïve Bayes classifier	[19]
	Mohler dataset and (CU)-NLP dataset	MaLSTM	[20]
Named Entity Recognition	Logistics Vocational Education	Siamese Neural Network	[21]

Text Classification

Text classification, also known as document categorization, is the most in-demand area of ML for understanding texts written in any natural language [22]. It is a fundamental technique in NLP that involves assigning predefined categories to text documents [22]. According to Shaik, Tao, Li, Dann, McDonald, Redmond and Galligan [13], text classification plays a crucial role in evaluating and scoring student responses.

Crossley, Allen, Snow and McNamara [14] study employed text classification techniques to analyze essays on multiple linguistic levels. Specifically, they extracted a range of linguistic features, including syntactic complexity, lexical sophistication, cohesion, and textual relevance. They also used semantic analysis to gauge essay content, employing methods to evaluate coherence and relevance to the essay prompt. Features from Coh-Metrix, a text analysis tool that measures cohesion and readability, were incorporated to capture these linguistic elements and quantify aspects of essay quality. For their dataset, Crossley et al. used a collection of essays sourced from the Writing Pal (W-Pal), an intelligent tutoring system designed to support students in writing improvement. The study utilized regression-based ML algorithms to build predictive models for essay scoring.

Similarly, Jing, Santos, Boticario, Romero, Pechenizkiy and Merceron [3] explored the use of a semi-supervised document clustering technique called Constrained K-Means clustering for automatically grading short-answer responses in MOOCs. This technique involved NLP for text similarity measurement and feature extraction, helping the system group semantically similar answers together. The clustering technique allowed for the efficient handling of short texts, typical of MOOC answers, by focusing on key patterns and topics within the responses. The dataset for this study was collected from actual MOOC platforms containing student responses to short-answer questions. While specific details about the exact size or source are not fully elaborated, the dataset included answers that varied in quality and relevance, providing a realistic sample of responses for a MOOC grading system.

Likewise, Balaha and Saafan [4] introduced an automated framework designed to evaluate different types of questions, specifically multiple-choice questions (MCQs), essays, and equations. The study used text classification and semantic analysis to evaluate the essay-type answers. Semantic similarity measures were applied to assess the relevance of student responses to predefined correct answers, especially in the short and essay-style questions. It compared the different approaches to convert (i.e. embed) textual data such as essays and short answers into numerical data. The Word2Vec, FastText, Glove, and Universal Sentence Encoder (USE) were used as the embedding pre-trained models in the experiments. The expression-matching experiments were performed on a dataset (named HMB-EMD-v1) compiled by the authors.

Sakaguchi, Heilman and Madnani [15] also applied text classification to assess the correctness and quality of short answers. Their model used a variety of NLP features to determine how well student responses matched reference answers and evaluated them on this basis. The authors framed the task as a text classification problem where each answer is classified along a scoring scale. They did this by extracting features from student answers that were then fed into a ML classifier. Specifically, the model incorporated lexical, syntactic, and semantic features (such as Bag-of-Words, dependency parsing, and Latent Semantic Analysis) to improve its classification ability. Using Support Vector Machines (SVMs), they could accurately assign scores based on the presence of relevant terms, grammatical structure, and semantic similarity to model answers. They conducted the experiments on short-answer questions that were developed under the Reading for Understanding (RfU) dataset.

Xie, Niu, Xue and Guan [16] explored an innovative approach to automated grading using Large Language Models (LLMs) in a three-stage system designed to better emulate human grading standards. Their system comprised rubric generation, grading, and post-grading review, each stage enhanced by specific NLP techniques and multi-agent collaboration among LLMs. To achieve human-like grading quality, the system utilizes one-shot prompts, self-reflection prompts, and batching prompts. These prompts allow LLMs to learn from examples (one-shot), refine answers iteratively (self-reflection), and ensure consistency across responses (batching). The Score-Distribution-Aware Sampling method ensures that sampled student answers reflect a diverse range of scores, which helps the LLMs understand varied student performance under the grading rubric. In addition, an open-sourced (OS) dataset from an undergraduate operating systems course was utilized. They implemented the system and evaluated it on both the OS dataset and the Mohler dataset. Experimental results showed significant performance improvements with the system.

Text Summarization

Text summarization techniques provide a summary of student feedback or a corpus of feedback text without losing critical information [13]. This process can be achieved through extractive, abstractive, or hybrid methods [23]. The extractive text summarization approach selects the most important sentences from the input document(s) and composes the output summary by combining these selected sentences. It uses feature extraction methods such as Bag of Words (BoW), N-grams, graphs, and similar approaches to select a subset of the sentences in an original text. The abstractive text summarization approach creates an intermediate representation of the input document(s) and generates the output summary from this representation. Unlike extractive summaries, abstractive summaries consist of sentences that are different from those in the original document(s). The hybrid text summarization approach combines elements of both extractive and abstractive methods.

Royan, Jayakumaran and Stephan [17] study presents a hybrid model utilizing text summarization techniques for the automated grading of descriptive assignments. The approach uses a dataset specifically compiled from descriptive responses in an educational setting, demonstrating the process of automating grading by aligning student responses with a baseline reference answer. This alignment reflects real-world scenarios where assessments, such as assignments and classroom tests, support student learning by providing structured feedback. The study implemented a combination of the RAKE (Rapid Automatic Keyword Extraction) and ROUGE (Recall-Oriented Understudy for Gisting Evaluation) algorithms to identify key phrases and score responses based on their relevance to the reference answers. This method aims to streamline grading and offer consistent, meaningful feedback to enhance student engagement and comprehension.

Agarwal, Kalia, Bahel and Thomas [18] developed an automated system called Auto-Eval: Automatic Test Evaluation System, which leverages NLP algorithms for grading. Their approach centers on an algorithm that assesses theoretical answers by matching key terms and computing a similarity index. This system aims to streamline grading, reduce manual effort, and increase the speed of evaluation. During the exam process, the staff creates an answer key for the question paper, which is stored in a database. Student responses are then compared against these reference answers, and the NLP algorithms

evaluate the responses by checking for accurate meaning, spelling, and grammar, as well as calculating the similarity index. Important metrics such as answer length and key term usage are also factored into scoring. The algorithms used have demonstrated accuracy and efficiency in grading. For text preprocessing, they utilized the NLTK library, performing word and sentence tokenization, along with applying the Gensim package and term frequency-inverse document frequency (TF-IDF) methods for final mark evaluation. In addition, they incorporated a Recursive Neural Network (RNN) model for training to handle various handwriting styles.

Sentiment Annotation

Sentiment annotation is one of the most popular types of annotations in NLP [13]. This process involves labelling the emotions, opinions, and sentiments expressed in a text, which can be categorized as positive, neutral, or negative sentiment [24]. According to Shaik, Tao, Li, Dann, McDonald, Redmond and Galligan [13], sentiment annotation techniques are employed to analyze and interpret students' written responses, feedback, or essay content.

Sadanand, Guruvyas, Patil, Acharya and Suryakanth [19] developed a model to automatically grade student essays by leveraging NLP and sentiment analysis techniques. This system employs an LSTM network model trained on a dataset of approximately 12,000 essays. Sentiment analysis is performed to determine the sentiment of the essay as either positive, negative or neutral. The Twitter sample dataset is used to build a sentiment classifier that analyzes the sentiment based on the student's approach towards a topic. Additionally, each essay is subjected to the detection of syntactical errors as well as plagiarism checks to detect the novelty of the essay. The model's performance was assessed using the Quadratic Weighted Kappa (QWK) metric, achieving an impressive average score of 0.911. Additionally, a Naïve Bayes classifier was used to construct a sentiment classifier, which determines whether students exhibit a positive or negative stance on a given topic. This classifier demonstrated an accuracy rate of 99.4%.

Tulu, Ozkaya and Orhan [20] propose an automated approach to evaluate short answers using a Manhattan LSTM (MaLSTM) network trained on two datasets: the widely used Mohler dataset, a benchmark in Automatic Short Answer Grading (ASAG) research, and the Cukurova University (CU)-NLP dataset, created specifically for this task by the Department of Computer Engineering at Cukurova University. A key innovation in this work is its use of semantic sense representations generated from concepts within the WordNet lexical-semantic network, employing the SemSpace method. SemSpace generates vectors for each word sense based on WordNet synsets and their interrelations, offering a richer semantic understanding for grading. Training with the Mohler dataset yielded promising results, demonstrating both reliability and consistency. The CU-NLP dataset further validated the approach, proving its adaptability to domain-specific requirements. Notably, the effectiveness of the proposed model can be attributed to both the SemSpace sense vectors and the architecture of the MaLSTM network.

Named Entity Recognition

Named Entity Recognition (NER) stands as a cornerstone task in NLP, possessing immense significance in information extraction and knowledge organization [13]. NER

involves the identification and classification of named entities, such as names of individuals, organizations, locations, dates, and more, within textual data. Its pivotal role lies in its capacity to disentangle structured information from unstructured text, thereby enhancing data retrieval, analysis, and understanding.

Xie and Chen [21] utilize NER as part of the model’s NLP pipeline to improve the accuracy of scoring subjective answers. NER helps the model identify and extract key entities such as names, dates, locations, and specialized terminology from students’ responses. With the framework of deep learning, they use the simulated manual scoring method to extract and match the score point sequence of students’ answers with standard answers. The score recognition model effectively improves the efficiency of model calculation and long-text keyword matching. The loss value of the final training score recognition model is about 0.9, and the accuracy is 80.54%. The accuracy of the training text similarity matching model is 86.99%, and the fusion model is single. The scoring time is less than 0.8s, and the accuracy is 83.43%.

3.2 Research Outcomes of Automated Assessment System

RQ2 investigated the research aims/objectives and major findings of automatic grading of assessment systems. To explore the research outcomes for automated assessment systems, we identified how each paper reported the key results of model performance. These findings are summarized in Table 4.

Table 4. Reviewed papers on research outcomes of automated assessment systems

Author	Enhanced Efficiency & Consistency	Improved Accuracy & Reliability	Sentiment & Emotion Analysis	Scalability & Accessibility
[3]	✓			✓
[16]		✓		
[18]	✓			
[25]	✓			✓
[19]		✓	✓	
[20]			✓	

Enhanced Efficiency and Consistency

Huawei and Aryadoust [25] study provides an in-depth review of automated essay scoring (AES) systems, evaluating their performance, validity, and application in educational settings. It outlines the evolution of AES technologies and their impact on educational practices. Key outcomes include improvements in scoring consistency, the ability to handle large volumes of student work efficiently, and the potential for providing immediate feedback to learners. Similarly, Jing, Santos, Boticario, Romero, Pechenizkiy and Merceron [3] study highlights how the K-means clustering technique has efficiently handled

short texts, typical of MOOC answers, by focusing on key patterns and topics within the responses. Likewise, Agarwal, Kalia, Bahel and Thomas [18] highlight that the score recognition model effectively improves the efficiency of model calculation and long-text keyword matching, as well as feedback and scoring, thereby reducing the subjectivity inherent in human grading.

Improved Accuracy in Grading

Many automated assessment systems have demonstrated high accuracy levels, sometimes comparable to human graders. For example, a study on “Grade Like a Human” outlines a multi-agent system designed to refine rubrics and improve grading accuracy through iterative self-reflective learning paradigms [16]. This system ensures that scoring criteria are optimized based on actual student responses, leading to more precise and reliable grading outcomes. Sadanand, Guruvyas, Patil, Acharya and Suryakanth [19] study also integrated NLP techniques with sentiment analysis in an automated grading system achieving improved grading accuracy and consistency.

Sentiment and Emotion Analysis

Incorporating sentiment analysis techniques allows for a deeper understanding of student responses beyond just correctness [20]. For example, Sadanand, Guruvyas, Patil, Acharya and Suryakanth [19] use sentiment analysis to identify the tone, sentiment, and emotional nuances expressed in students’ writing. This layer of analysis allows the system to detect whether the content aligns with the expected emotional and attitudinal response required by the essay prompt.

Scalability and Accessibility

According to Huawei and Aryadoust [25], automated systems can scale to accommodate large numbers of students, making them ideal for MOOCs and other large-scale educational settings. Their study’s approach using semi-supervised document clustering allows the system to process and evaluate answers in bulk without needing each response to be individually assessed by human graders. By clustering similar answers, the model can assign scores based on clustered patterns, which is a scalable solution for handling thousands of student responses [3].

3.3 Future Directions on NLP Automatic Grading of Assessments

RQ3 explored the future directions for the automatic grading of assessments. While significant progress has been made, several areas offer promising avenues for future research on NLP and automatic grading of assessments. These future works are summarized in Table 5.

Explainable Artificial Intelligence (XAI) is crucial for building trust and transparency in automated assessment systems [26, 27]. It helps to provide clear explanations for the grades assigned by automated systems, enhancing trust and acceptance among educators and students [28]. According to Arrieta, Díaz-Rodríguez, Del Ser, Bennetot, Tabik, Barbado, García, Gil-López, Molina and Benjamins [29], integrating XAI into automated grading systems can address concerns about bias, fairness, and accountability, ensuring that AI-driven evaluations are both accurate and transparent.

Table 5. Future work on NLP and automatic grading of assessments

Author	XAI	Adaptive Assessment	Multimodal Assessment	Authentic Assessment	Deep Learning Integration
[26]	✓				
[27]	✓				
[28]	✓				
[29]	✓				
[30]		✓			
[31]		✓			
[32]		✓			
[33]		✓			
[34]			✓		
[35]			✓		
[36]			✓		
[37]				✓	
[38]				✓	
[39]				✓	✓
[40]				✓	
[41]				✓	
[42]					✓
[43]					✓
[44]					✓
[45]					✓
[46]					✓

Adaptive assessment refers to the use of algorithms and ML techniques to create dynamic and personalized testing experiences that adapt to the learner's level of knowledge and skills [30, 31]. This approach enhances the accuracy and efficiency of assessments, ensuring that each student is evaluated fairly based on their individual learning journey [32]. Incorporating adaptive assessment techniques can significantly improve the educational experience by providing personalized, real-time feedback and support [33]. Future work should focus on integrating these systems with existing educational platforms, expanding their capabilities to multi-modal assessments, and ensuring ethical considerations are addressed.

Multimodal assessment refers to the evaluation of student performance through multiple forms of data, such as text, speech, and video, to provide a comprehensive understanding of their abilities and knowledge [34, 35]. Integrating multimodal assessment analysis to assess various forms of student expression can enhance the accuracy,

depth, and fairness of evaluations, capturing a broader range of student interactions and responses [36].

Authentic assessment refers to evaluating students' abilities through real-world tasks and activities that reflect how their knowledge and skills will be used outside the academic environment [37, 38]. In the context of NLP in automatic grading of assessments, incorporating authentic assessment involves leveraging NLP techniques to analyze complex, open-ended student responses that simulate real-world scenarios [39]. This approach emphasizes the application of knowledge in practical contexts, fostering deeper learning and critical thinking [40]. Future work should aim to develop advanced NLP models, integrate real-world data, address ethical considerations, and ensure scalability to enhance the impact and effectiveness of authentic assessments [41].

Deep Learning Integration has significantly transformed various fields, including NLP and educational assessment [42, 43]. Its integration into automated grading systems promises to enhance the accuracy, consistency, and efficiency of evaluations [39]. Future work should focus on leveraging advanced deep learning techniques such as BERT (Bidirectional Encoder Representations from Transformers) and GPT-3 (Generative Pre-trained Transformer 3) for enhanced text understanding and generation [44–46].

3.4 Limitations of NLP Techniques

The lack of a distinct measure for analytic, text-based writing hinders the researchers' ability to define effective literacy practices, as there are no adequate measures of student reasoning and analytic ability that can be correlated with theoretical measures of effective teaching [47]. Similarly, while NLP models can provide grades, they often struggle to offer clear, interpretable feedback that would help students improve [1]. Many models lack transparency in their decision-making processes, and explaining the rationale behind grades remains a challenge, particularly in more advanced models like deep neural networks. Moreover, NLP models trained on biased datasets can reflect and amplify these biases in their grading, affecting their fairness and reliability [48]. For instance, if a grading system is trained on data that inadvertently favors certain dialects or expressions, it may unfairly penalize students from diverse linguistic backgrounds [48]. Bias mitigation remains an ongoing area of research for NLP, and achieving a truly equitable grading system is still a significant challenge.

4 Conclusion and Limitations of the Study

This SLR enhances our understanding of NLP in the automatic grading of assessments by providing an updated overview of the current state of research in this area and suggesting potential avenues for future investigations. NLP has emerged as a transformative technology in the realm of educational assessments, providing enhanced accuracy, consistency, and efficiency in grading. The integration of NLP techniques in automatic grading systems addresses several traditional challenges, such as subjective biases, time constraints, and the need for scalable solutions in large educational settings. As technology continues to evolve, its integration with advanced AI models, ethical frameworks, and personalized learning approaches will further solidify its role in shaping the future

of education. The ongoing research and development in this area promise to create more robust, fair, and effective assessment systems that cater to the diverse needs of learners worldwide.

However, the limitation of the study is the possibility of overlooking relevant research due to database constraints and biases in data extraction. While Scopus was used, other databases might contain pertinent studies. This could mean that other relevant studies are missing, thus limiting the generalizability of findings and potentially skewing interpretations of trends within the NLP domain. Likewise, the bias in data extraction may lead to overemphasis on certain methodologies or findings that are more commonly reported in Scopus, potentially giving a false impression of the field's dominant approaches or conclusions. This knowledge gap could impact the deployment of NLP applications in higher education. Nevertheless, given the comprehensive nature of the initial search within the Scopus database, the study offers valuable insights into the domain's key trends. Future research should explore other databases to diversify the datasets and enhance the generalizability of NLP systems. Additionally, future studies should conduct qualitative exploration of the user experience and practical implications of NLP systems in educational contexts. Gathering insights from both educators and students on the integration of these systems could offer a more rounded perspective.

References

1. Alqahtani, T., Badreldin, H.A., Alrashed, M., Alshaya, A.I., Alghamdi, S.S., bin Saleh, K., Alowais, S.A., Alshaya, O.A., Rahman, I., Al Yami, M.S.: The emergent role of artificial intelligence, natural learning processing, and large language models in higher education and research. *Research in Social and Administrative Pharmacy*, **19**(8), 1236–1242 (2023)
2. Burrows, S., Gurevych, I., Stein, B.: The eras and trends of automatic short answer grading. *Int. J. Artif. Intell. Educ.* **25**, 60–117 (2015)
3. Jing, S., Santos, O., Boticario, J., Romero, C., Pechenizkiy, M., and Merceron, A.: Automatic Grading of Short Answers for MOOC via Semi-supervised Document Clustering. In: EDM, pp. 554–555 (2015)
4. Balaha, H.M., Saafan, M.M.: Automatic exam correction framework (AECF) for the MCQs, essays, and equations matching. *IEEE Access* **9**, 32368–32389 (2021)
5. Rokade, A., Patil, B., Rajani, S., Revandkar, S., and Shedge, R.: 'Automated grading system using natural language processing', In: 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, India, pp. 1123–1127. (2018). <https://doi.org/10.1109/ICICCT.2018.8473170>
6. Ramesh, D., Sanampudi, S.K.: An automated essay scoring systems: a systematic literature review. *Artif. Intell. Rev.* **55**(3), 2495–2527 (2022)
7. Gao, R., Merzdorf, H.E., Anwar, S., Hipwell, M.C., Srinivasa, A.: Automatic assessment of text-based responses in post-secondary education: A systematic review. *Computers and Education: Artificial Intelligence*, p. 100206 (2024)
8. Mallett, R., Hagen-Zanker, J., Slater, R., Duvendack, M.: The benefits and challenges of using systematic reviews in international development research. *J. Dev. Effectiv.* **4**(3), 445–455 (2012)
9. Sarkis-Onofre, R., Catalá-López, F., Aromataris, E., Lockwood, C.: How to properly use the PRISMA Statement. *Syst. Rev.* **10**, 1–3 (2021)

10. Schotten, M., Meester, W.J., Steinginga, S., and Ross, C.A.: A brief history of Scopus: The world's largest abstract and citation database of scientific literature: 'Research analytics, Auerbach Publications, pp. 31–58 (2017)
11. Zupic, I., Čater, T.: Bibliometric methods in management and organization. *Organ. Res. Methods* **18**(3), 429–472 (2015)
12. Albhirat, M.M., Rashid, A., Rasheed, R., Rasool, S., Zulkiffli, S.N.A., Zia-ul-Haq, H.M.: 'The PRISMA statement in enviropreneurship study: a systematic literature and a research agenda', *Cleaner Engineering and Technology*, p. 100721 (2024)
13. Shaik, T., et al.: A review of the trends and challenges in adopting natural language processing methods for education feedback analysis. *IEEE Access* **10**, 56720–56739 (2022)
14. Crossley, S.A., Allen, L.K., Snow, E.L., McNamara, D.S.: Incorporating learning characteristics into automatic essay scoring models: What individual differences and linguistic features tell us about writing quality. *J. Educ. Data Mining* **8**(2), 1–19 (2016)
15. Sakaguchi, K., Heilman, M., Madnani, N.: 'Effective feature integration for automated short answer scoring', In: *Proceedings of the 2015 conference of the North American Chapter of the association for computational linguistics: Human language technologies*, pp. 1049–1054. (2015)
16. Xie, W., Niu, J., Xue, C.J., Guan, N.: Grade Like a Human: Rethinking Automated Assessment with Large Language Models' (2024). arXiv preprint [arXiv:2405.19694](https://arxiv.org/abs/2405.19694)
17. Royan, R., Jayakumaran, C., Stephan, T.: 'Text summarization for automatic grading of descriptive assignments: A hybrid approach': 'Enterprise Digital Transformation' Auerbach Publications, , pp. 251–273. (2022)
18. Agarwal, M., Kalia, R., Bahel, V., Thomas, A.: 'Autoeval: A NLP approach for automatic test evaluation system', In: *2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCon)*, Kuala Lumpur, Malaysia, pp. 1–6. (2021). <https://doi.org/10.1109/GUCon50781.2021.9573769>
19. Sadanand, V.S., Guruvyas, K.R.R., Patil, P.P., Acharya, J.J., Suryakanth, S.G.: An automated essay evaluation system using natural language processing and sentiment analysis. *Int. J. Electr. Comp. Eng.* **12**(6), 6585–6593 (2022)
20. Tulu, C.N., Ozkaya, O., Orhan, U.: Automatic short answer grading with SemSpace sense vectors and MaLSTM. *IEEE Access* **9**, 19270–19280 (2021)
21. Xie, B., Chen, L.: 'Automatic Scoring Model of Subjective Questions Based Text Similarity Fusion Model', In: *International Conference on Wireless Communications, Networking and Applications*, pp. 586–599. Singapore: Springer Nature Singapore (2021)
22. Dogra, V., Verma, S., Kavita, Chatterjee, P., Shafi, J., Choi, J., Ijaz, M.F.: A Complete Process of Text Classification System Using State-of-the-Art NLP Models. *Comp. Intell. Neurosci.* **2022**(1), 1883698 (2022)
23. El-Kassas, W.S., Salama, C.R., Rafea, A.A., Mohamed, H.K.: Automatic text summarization: A comprehensive survey. *Expert Syst. Appl.* **165**, 113679 (2021)
24. Sutoyo, E., Almaarif, A., Yanto, I.T.R.: Sentiment analysis of student evaluations of teaching using deep learning approach, In: *The International Conference on Emerging Applications and Technologies for Industry 4.0*, pp. 272–281. Cham: Springer International Publishing (2020)
25. Huawei, S., Aryadoust, V.: A systematic review of automated writing evaluation systems. *Educ. Inf. Technol.* **28**(1), 771–795 (2023)
26. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.* **267**, 1–38 (2019)
27. Hall, E., Seyam, M., Dunlap, D.: 'Exploring Explainability and Transparency in Automated Essay Scoring Systems: A User-Centered Evaluation', In: *International Conference on Human-Computer Interaction*, pp. 266–282. Cham: Springer Nature Switzerland (2024)

28. Wang, D., Yang, Q., Abdul, A., Lim, B.Y.: ‘Designing theory-driven user-centric explainable AI’, In: Proceedings of the 2019 CHI conference on human factors in computing systems, pp. 1–15. (2019)
29. Arrieta, A.B., et al.: Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Inform. Fus.* **58**, 82–115 (2020)
30. Morze, N., Varchenko-Trotsenko, L., Terletska, T., Smyrnova-Trybulska, E.: Implementation of adaptive learning at higher education institutions by means of Moodle LMS. *J. Phys. Conf. Ser.* **1840**(1), 012062 (2021)
31. Khine, M.S.: Using AI for Adaptive Learning and Adaptive Assessment: Artificial Intelligence in Education: A Machine-Generated Literature Overview. Springer, pp. 341–466. (2024)
32. Smyrnova-Trybulska, E., Morze, N., Varchenko-Trotsenko, L.: Adaptive learning in university students’ opinions: Cross-border research. *Educ. Inf. Technol.* **27**(5), 6787–6818 (2022)
33. Xiaoyu, Z., Tobias, T.: Exploring the Efficacy of Adaptive Learning Technologies in Online Education: A Longitudinal Analysis of Student Engagement and Performance. *Int. J. Sci. Eng. Appl.* **12**(12), 28–31 (2023)
34. Ross, J., Curwood, J.S., Bell, A.: A multimodal assessment framework for higher education. *E-Learn. Digital Media* **17**(4), 290–306 (2020)
35. Dressen-Hammouda, D., Wigham, C.R.: Evaluating multimodal literacy: Academic and professional interactions around student-produced instructional video tutorials. *Syst.* **105**, 102727 (2022)
36. Baltrušaitis, T., Ahuja, C., Morency, L.-P.: Multimodal machine learning: A survey and taxonomy. *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(2), 423–443 (2018)
37. Sokhanvar, Z., Salehi, K., Sokhanvar, F.: Advantages of authentic assessment for improving the learning experience and employability skills of higher education students: A systematic literature review. *Stud. Educ. Eval.* **70**, 101030 (2021)
38. Ajjawi, R., Tai, J., Dollinger, M., Dawson, P., Boud, D., Bearman, M.: From authentic assessment to authenticity in assessment: broadening perspectives. *Assess. Eval. High. Educ.* **49**(4), 499–510 (2024)
39. Devlin, J.: Bert: Pre-training of deep bidirectional transformers for language understanding (2018). arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805)
40. McArthur, J.: Rethinking authentic assessment: work, well-being, and society. *High. Educ.* **85**(1), 85–101 (2023)
41. Salinas-Navarro, D.E., Vilalta-Perdomo, E., Michel-Villarreal, R., Montesinos, L.: Using generative artificial intelligence tools to explain and enhance experiential learning for authentic assessment. *Educ. Sci.* **14**(1), 83 (2024)
42. Torfi, A., Shirvani, R.A., Keneshloo, Y., Tavaf, N., Fox, E.A.: ‘Natural language processing advancements by deep learning: A survey’ (2020). arXiv preprint [arXiv:2003.01200](https://arxiv.org/abs/2003.01200)
43. Khan, W., Daud, A., Khan, K., Muhammad, S., Haq, R.: Exploring the frontiers of deep learning and natural language processing: A comprehensive overview of key challenges and emerging trends. *Nat. Lang. Process. J.* pp. 100026 (2023)
44. Kondurkar, I., Raj, A., Lakshmi, D.: Modern applications with a focus on training ChatGPT and GPT models: Exploring generative AI and NLP’: ‘Advanced applications of generative AI and natural language processing models, IGI Global, pp. 186–227. (2024)
45. Saylo, K.R., Saylo, M.: Innovative Content Generation: Leveraging GPT-3 Language Capabilities. *J. Innov. Technol. Converg.* **5**(2) (2023)
46. Anusha, B., Borah, A.R., Mishra, G., Thethi, H.P., Al-Allak, M.A., Vidhya, B.V.: Merging the Strengths of GPT-4 and DeepLab for Semantic Text Understanding in Digital Textbooks, In: 2024 International Conference on Communication, Computer Sciences and Engineering (IC3SE), pp. 1–6. IEE, (2024)
47. Correnti, R., Matsumura, L.C., Wang, E., Litman, D., Rahimi, Z., Kisa, Z.: Automated scoring of students’ use of text evidence in writing. *Read. Res. Q.* **55**(3), 493–520 (2020)

48. Dimari, A., Tyagi, N., Davanageri, M., Kukreti, R., Yadav, R., Dimari, H.: 'AI-based automated grading systems for open book examination system: implications for assessment in higher education', In: 2024 International Conference on Knowledge Engineering and Communication Systems (ICKECS), vol. 1, pp. 1–7. IEEE (2024)



An Attention-Based Deep Learning Model for Term Extraction from Text Using BERT

Tsitsi Zengeya¹(✉), Jean Vincent Fonou Dombou¹, and Mandlenkosi Gwetu²

¹ School of Mathematics, Statistics and Computer Science, University of KwaZulu
Natal, Pietermaritzburg 3209, South Africa

222131004@stu.ukzn.ac.za, fonoudombouj@ukzn.ac.za

² Department of Industrial Engineering, University of Stellenbosch, Stellenbosch
7600, South Africa

mgwetu@sun.ac.za

Abstract. Extracting terms from text is relevant in many applications such as document summarizing, question answering, ontology learning and many more. However, the unstructured nature of textual data poses significant hurdles that make the automatic extraction of terms from text a challenging task. Furthermore, while significant research efforts have been dedicated to term extraction from textual data, the incorporation of domain knowledge and context awareness as a way of enriching the extracted terms remains a challenge. To tackle these challenges, this study proposes an attention-based Deep Learning model for term extraction from text using the Bidirectional Encoder Representations from Transformers (BERT) language model. The model uses (1) web scraping for term extraction from web pages, (2) BERT encoder for enriching term extraction through contextualization and cosine similarity to extract domain-specific terms and (3) WordNet for adding knowledge context and disambiguation through vocabulary labeling. The proposed system was applied to five different data samples from the horticulture domain and was evaluated with various metrics including accuracy, precision, and F1-score. The experimental results indicate that the proposed model improves the domain-specific term extraction and vocabulary labeling with an accuracy of 73%, a precision of 89%, a recall of 79%, as well as F1-Score of 84%, better than the related studies.

Keywords: BERT · Term Extraction · WordNet · Vocabulary Building · Cosine Similarity Measure

1 Introduction

Term extraction is a foundational task with wide-ranging applications across various domains, among them document summarization, question answering, and ontology learning [1]. In the context of document summarization, term extraction involves identifying key terms to create brief summaries, allowing readers

to quickly grasp the main idea of a term and locating important information effectively. Term extraction helps with understanding user queries in question answering systems and delivering accurate responses through retrieval. At the same time, ontology learning also requires term extraction as one of the basic components because it contributes to developing structured representations of knowledge that reside in particular domains [2]. These ontologies serve as valuable resources for organizing domain-specific knowledge in applications like information retrieval and data integration.

While significant research efforts have been dedicated to understanding vocabulary acquisition from web pages using web scrapping, the incorporation of domain knowledge, and context awareness, language variation reconciliation as a way of enriching term extraction remains a challenge [3]. This difficulty arises because the incorporation of diverse linguistic forms, regional dialects, and contextual usages complicates the task of accurately identifying and extracting domain-specific terms. Consequently, the terms extracted may introduce new vocabulary that does not faithfully represent the intended domain, potentially leading to erroneous labeling or categorization. The inherent lack of structure in textual data further exacerbates these challenges, rendering traditional vocabulary extraction methods insufficient for comprehensive analysis. To address this complexity, it becomes increasingly essential to harness advanced computational techniques capable of navigating and processing unstructured textual data effectively. By leveraging methodologies such as natural language processing, deep learning, and semantic analysis, researchers can enhance the accuracy and relevance of term extraction, thereby facilitating more reliable insights and knowledge discovery within diverse domains.

Many studies on enriching term extraction, are using Natural Language Processing (NLP), Named Entity Recognition (NER), Topic Modeling, Word embedding, Keyword Extraction, Ontology-Based Methods, and Semantic Similarity [4]. However these methods fall short in various ways: Traditional word embeddings, such as Word2Vec or Global Vectors (GloVe) [5], offer static representations of words based on co-occurrence statistics, lacking contextual variations in word meanings [5]. This limitation hinders their ability to accurately represent word semantics across different contexts. Additionally, NLP techniques like NER and keyword extraction often lack context awareness, leading to inaccuracies in identifying important terms within specific contexts. Similarly, while topic modeling can uncover latent topics, it may overlook fine-grained semantic relationships between terms, resulting in extracted terms lacking semantic richness and precision. Pre-trained sentence transformers such as Generative Pre-trained Transformer (GPT-1) [6], BERT [7], Robustly Optimized BERT Pre-training Approach (Roberta) [8] and many more, address the issue of lack of semantic richness and precision by obtaining contextualized embeddings. A number of studies have proposed various term extraction solutions based on BERT [9, 10] [11] [12]. However, these term extraction methods do not fully utilize domain-specific knowledge or adequately account for contextual nuances in the text [13] [14]. Integrating such knowledge and awareness into automated term extrac-

tion systems requires advanced techniques and algorithms capable of effectively capturing and utilizing domain-specific and contextual information.

For instance developing new products such as fertilizer may require an understanding on the challenges being faced by farmers when growing vegetables. Manually analyzing large volumes of unstructured text data to understand these challenges is time-consuming and prone to missing important terms that identify specific plant names, challenges or farming methods mentioned by farmers can be difficult. To address these problems, techniques like BERT can be used for term extraction and text analysis in the context of vegetable farming. Cosine similarity can be used to score term similarities for the purposes of determining term relevance to given queries. WordNet can provide contextual knowledge clarifying related concepts. This enriched term extraction allows farmers to describe their challenges more accurately thus, contributing to the development of targeted fertilizer solutions for common issues. Extracted terms can also classify forum posts and discussions, helping in trend analysis, and identifying new problems. In horticulture, term extraction helps understand the specific language farmers use, yielding insights that guide the creation of better products.

This paper presents an attention-based deep learning model for term extraction from text with BERT. The BERT model takes processed terms as input, converts them to fixed embeddings, which are then passed on to the BERT Encoder to produce contextualized embeddings of the terms. The produced contextualized embeddings are then compared with the embeddings of predefined keywords using the cosine similarity. The terms with cosine similarity above the threshold are then passed through WordNet for knowledge context. WordNet [15] identifies synsets of terms and uses part-of-speech (POS) tagging to determine if the term is a verb, noun, or digit. This information is crucial for the production of a list of terms that may be used to build an ontology of the domain. The key contribution of the study lies in the enrichment of the term extraction process through contextualized term representation, extraction of domain-related terms after using the cosine similarity measures, and adding knowledge context through the use of WordNet. The whole process aids in building a more relevant and accurate vocabulary of terms with correct labels.

The rest of the paper is structured as follows. Section 2 discusses related research. Section 3 presents the materials and methods employed in the proposed model. Section 4 describes and discusses the experimental results as well as a comparison of our results with existing studies. Section 5 concludes the study and suggests potential avenues for future research.

2 Related Work

The automatic extraction of terms from text has been a subject of interest to many authors [5, 16–18]. The study by Yih et al. [19] proposed a supervised method for extracting keywords on webpages. The authors implemented a two-step method where specific rules were firstly used to extract potential keywords, then metrics like Term Frequency - Inverse Document Frequency (TF-IDF),

position, and structural/syntactic features were used to train a model to predict whether each candidate keyword is relevant or not. However, supervised methods require a large amount of labeled data which has to be thus manually tagged as keywords that are relevant or irrelevant. Such manual tagging is also sensitive to noise and outliers, resulting to possible model overfitting and ultimately inaccurate keyword extraction.

Mihalcea et al. [20] proposed a solution for document processing where text documents were treated as nodes in a graph and two nodes were connected if they exhibited high correlations. The edge weight was determined by the number of outgoing edges from each node. Within the text graph, real values were incorporated as ranks, and statistical calculations were applied iteratively until the rank values converged. Although graph-based approaches can identify terms based on their connections in a network, they struggle to understand the deeper meaning or context of the terms, leading to the extraction of irrelevant or non-representative terms.

A study by Lu et al. [9] proposed an Entity-BERT model for extracting medical terms from Electronic Medical Records. The study used cross-attention to handle the complex and diverse language expressions and the experimental results show that the model performed well on entity recognition tasks. Another study [17] employed the BERT model for biomedical knowledge discovery from an unstructured text corpora using contextual word embeddings. The study compared the produced contextualized word embeddings with the static equivalent embeddings from static models such as Word2Vec. Sharma et al. [18] proposed a self-supervised contextualized keyword and keyphrase retrieval with self-labelling to extract keywords. The model employed BERT for contextual features while long short-term memory (LSTM) was used for labeling. While these studies look similar, our research differs by adding category-specific layers to BERT encodings to better extract domain-specific terms and later uses WordNet to obtain contextual knowledge. Our approach not only improves the accuracy of term extraction but also enables a better understanding of the underlying semantic structure within specific domains.

Other relevant studies by Devika et al. [10], Vailhya et al. [11] and Zang et al. [12] have proposed various solutions for term extraction from textual data based on BERT model. In the study by Devika et al. [10] the authors extracted key phrases from five different datasets using their proposed semkey-BERT model. The model was able to retain semantic and syntactic relationships among tweets. However, the proposed model only considers 3-gram key phrases, that is the proposed model only considers key phrases that are exactly three words long in terms of size. This poses a limitation because the size of n-grams can impact on the quality and relevance of extracted key phrases. By imposing a restriction of 3-grams, the model may miss out on important key phrases that are shorter or longer than three words. Also their method did not experiment with different hyperparameters to find the optimal configuration for the proposed model [10]. Zang et al. [12], used BERT for information extraction on domain-specific business documents. The research was treated as a sequence labeling task, and

BERT was used to extract content elements from two different type of business documents. All the documents were segmented into paragraphs and processed at the paragraph level. A fully-connected layer with softmax activation were added on top to perform the predictions. In the study by Vailhya et al. [11], the authors used domain-specific BERT models for Named Entity Recognition (NER) tagging. The SciBERT, BioBERT, BERT-base, BERT-large were used to extract contextualised representation. A simple linear activation function coupled with a softmax layer served as a classifier to obtain probabilities on the labels from the contextualised representation. Although, this research and our research use BERT models they differ in their categorisation methods. The approach by Vailhya et al. [11] uses a linear classifier with softmax where classification is done directly by assigning probabilities that reflect their relevance while our research used cosine similarity with threshold to compute semantic similarity between terms and predefined categories which act as the reference points. Terms will be selected based on predefined similarity threshold, focusing on how closely terms relate to domain-specific references.

These studies highlight BERT’s ability to capture the contextual relationships between words, making it a powerful tool for tasks that require a deep understanding of language [1]. By utilizing pre-trained models like BERT and its derivatives, researchers and practitioners can significantly enhance the accuracy of term extraction processes [1]. These models not only handle the complexity of human language more effectively but also reduce the need for extensive manual annotation, thereby streamlining the development of NLP applications [1].

3 Proposed Attention-Based Deep Learning Model

This section describes our proposed attention-based Deep Learning model. Figure 1 shows the structure of the proposed model, which comprises content and term extraction phases. Texts are obtained from web pages and Scrapy framework was used to scrap HTML contents from web pages in stage I. In stage II, the BERT encoder is used to enrich term extraction, where a category-specific layer is placed on top of the BERT Encoder to extract terms based on a predefined category. The enriched term extraction is then evaluated to check the accuracy of the model. Wordnet is further applied for the identification of terms in the form of verbs, nouns and numerical values. The next subsections provide a more detailed description of the proposed attention-based Deep Learning model.

3.1 Preprocessing

The preprocessing consisted of extracting HTML content from the textual contents of web pages. The extraction of HTML content from the vast unstructured contents of web pages was firstly done through web scraping. The established web scraping framework, Scrapy [21], was used for this task. We began by identifying relevant URLs. Scrapy was then used to parse the content of these webpages

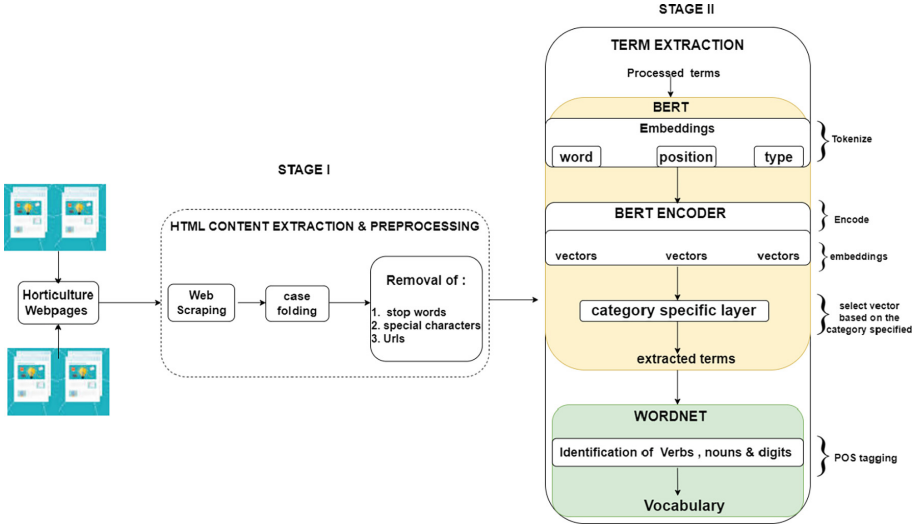


Fig. 1. Architecture of the Proposed attention-based Deep Learning Model.

and extract relevant terms based on predefined categories and its built-in features. To clean and normalize the extracted sentences, we preprocessed them by removing stop words, URLs, punctuation, tags, and special characters, and converting them to lowercase. The Natural Language Toolkit (NLTK), a Python library for text preprocessing, was employed for this purpose. NLTK enhances tokenization and analysis as well as improving model performance through noise reduction. Additionally, NLP text processors like tokenization, stemming, and lemmatization were utilized during preprocessing.

3.2 Enriched Term Extraction

To enrich terms extracted using web scraping, the preprocessed text is passed to the BERT model (Stage II in Fig. 1) together with predefined categories of texts to generate fixed-dimensional embeddings. The fixed-dimensional embeddings are then fed into a BERT Encoder. The architecture of the BERT Encoder is depicted in Fig. 2. BERT operates on a specific data format requiring special tokens ([CLS] for start and [SEP] for end) to delineate sentences. So, the text undergoes tokenization to align with BERT's vocabulary. Each tokenized phrase is assigned input ids, numerical identifiers linking them to indices in BERT's tokenizer vocabulary, resulting in vectors of 768 elements. These vectors, serving as contextual word embeddings, encapsulate the numerical representation of individual tokens. As a result, we obtain a tensor sized 768 by the number of tokens, with each token corresponding to a vector generated by the encoders. Leveraging these tensors, semantic representations of the input sequence were generated through transformation. Each word was analyzed in the self-attention layer on how it relates to all other words in the phrase, enabling the model

to capture long-range dependencies and contextual understandings. The output from the self-attention layer of the BERT Encoder in Fig. 2 is added back to the original word embedding and normalized in the Add and Normalize layer. The output is then passed to the feed-forward network layer where non-linear relationships between words are learnt.

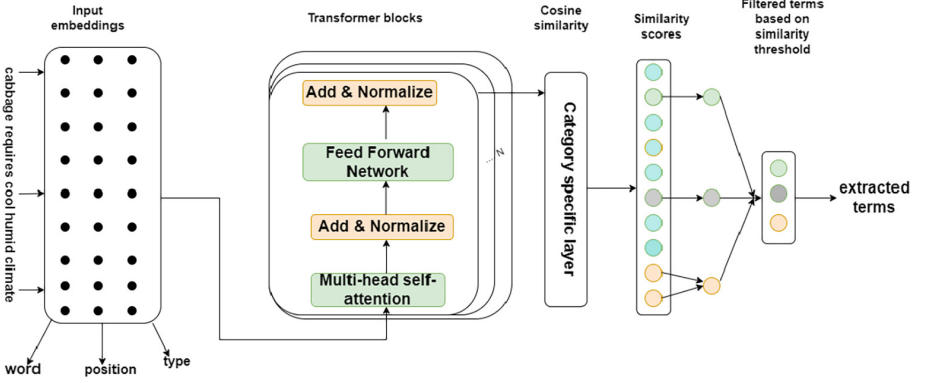


Fig. 2. BERT Encoder Architecture.

The contextualized embeddings for the terms from the five data samples (encoded $S_i, 1 \leq i \leq 5$), initially scraped and processed from the Food and Agriculture Organization (FAO) webpages, are used alongside the contextualized embeddings of predefined categories to compute semantic similarity using cosine similarity. These similarities guide the extraction of relevant, category-specific terms within the category-specific layer, positioned on top of the BERT encoder. This layer applies a targeted selection mechanism that filters and refines terms specifically related to horticulture. Trained with labeled data highlighting category-relevant terms, the layer enhances accuracy and context-specificity in term extraction. By prioritizing relevant terms and filtering out irrelevant ones, the category-specific layer fine-tunes the BERT embeddings with the following settings: learning rate of $1e - 5$, batch size of 4, over 5 epochs, optimizing the model for domain-specific applications. Terms that exhibit the highest similarity above a threshold of 0.7 to predefined category terms are identified as most relevant. Cosine similarity is chosen to measure the relationship between the predefined categories and the processed text due to its effectiveness in handling high-dimensional data. The cosine similarity between the vector representation of the processed text and the embeddings of the category-specific keywords is then calculated using Eqs. 1 and 2 to finalize the extraction.

$$\text{Cos}\theta = \frac{A.B}{\|A\|.\|B\|}, \quad (1)$$

$$= \frac{\sum_{i=1}^n A_i \cdot B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}, \quad (2)$$

where A is the vector representation of the processed text while B is the word embedding of the category keyword. Similarity scores were produced which reflect how semantically close each text was to the desired category. Terms with a similarity score above similarity threshold of 0.7 were filtered and constituted our extracted terms. The extracted terms were then passed on to the WordNet for part-of-speech tagging and we managed to identify verbs, noun, and numerical values which constituted our vocabulary to be used in the next task of ontology construction.

3.3 WordNet

The extracted terms were examined for part-of-speech tagging using WordNet. WordNet uses different algorithms for making synsets and linking them [15], this helps to create synsets which arrange words into a single concept and contain words that have similar meanings, these can be verbs, nouns, or adjectives as shown in Fig. 3. These synsets are used by the WordNet in order to tag terms as either verbs or nouns with others having semantic relationships like hyponymy, hypernymy, and meronymy in place. WordNet is sensitive to context and meaning of words when tagging terms in order to assign them the correct part of speech [22]. For example, in the sentence “cabbage requires cool humid climate,” WordNet identifies “cabbage” as a noun (NN), “requires” as a verb (VBZ, 3rd person singular present), and “cool” as an adjective (JJ), these synsets are linked by semantic relations like hypernymy, hyponymy and meronymy [22]. This step is crucial because WordNet doesn’t just rely on basic word matching but uses semantic information from synsets to accurately assign tags, ensuring that even ambiguous words are correctly classified based on their meaning within a sentence. Figure 3 shows this procedure where in each sentence a specific tag/label, indicating a particular speech element was used for all the words. These semantic relationships are based on the strong organization of synsets in WordNet.

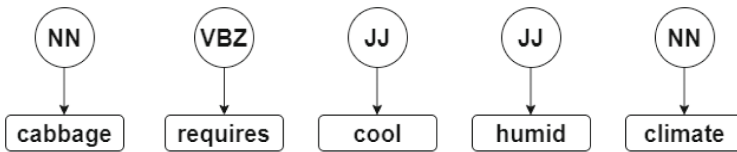


Fig. 3. example of Part-of-Speech tagging.

3.4 Performance Evaluation

The proposed term extraction method was evaluated using the following metrics: accuracy, precision, recall and F1-score as defined in Eqs. 3 to 6. Recall is regarded as the most important metric for term extraction, since high recall means missing fewer positive results [23].

$$Accuracy = \frac{TP}{TP + FP + TN + FN}, \quad (3)$$

where TP, TN, FP, and FN are true positive, true negative, false positive and false negative, respectively [24]. The precision examines how accurate a model's positive predictions are by giving a percentage of accurately extracted terms relative to the total number of extracted terms as in Equation (4).

$$Precision = \frac{TP}{TP + FP}. \quad (4)$$

Recall gives the percentage of the accurately extracted terms among all terms anticipated to be extracted [23] thus measuring completeness, as defined in Equation (5).

$$Recall = \frac{TP}{TP + FN}. \quad (5)$$

The F1-score is calculated as the harmonic mean of both precision and recall [16].

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}. \quad (6)$$

4 Results and Discussion

4.1 Dataset

This study used self-produced data samples (encoded S_i , $1 \leq i \leq 5$) obtained from the Food and Agriculture Organization of the United Nations (FAO) webpages. We collected terms from the web using the Scrapy framework [21]. The baseline dataset was constructed using stratified sampling where we got portions from each of the five classes to include in the baseline dataset. We employed stratified sampling to make sure that every sample in the baseline dataset was represented proportionally to its occurrence in the entire dataset. To be more precise, the dataset consisted of five samples, so we divided the dataset by calculating the percentage each sample represents in the total population. We then sampled each sample based on these proportions to maintain this distribution within the baseline dataset. For example samples S_1 and S_4 comprises 15% of the total data so 15% of the baseline was sampled from samples S_1 and S_4 respectively. Samples S_2 and S_3 comprises 25% of the total data so 25% of the baseline was sampled from sample S_2 and S_3 respectively. Sample S_5 comprises 20% of the total data so 20% of the baseline was sampled from sample S_5 . This method

maintained a relative representation of each class. Thus ensuring the baseline dataset is objective and representative of the sample distribution of the entire dataset. The baseline dataset can be accessed at [25]. To create a labeled dataset for model training, domain experts manually annotated the sentences in the data samples with multiple categories to ensure accurate and context-sensitive labeling. One or more labels were applied to each sentence, representing the variety of topics discussed in the text. The labels were selected using important agricultural topics that were pertinent to our research such as irrigation methods, soil conditions, water requirements, crop characteristics, climate conditions, effects of water supply, irrigation efficiency, fertilizer requirements, plant, and production. The multiple approach allows us to capture the complex, overlapping themes present in agricultural text, providing a comprehensive, context-rich dataset for model training. We divided the data into training, validation and test sets, split into 7:2:1 ration using scikitlearn library.

4.2 Total Number of Extracted Terms

After HTML content extraction using web scraping from the FAO website containing data about horticulture, 8900 terms including numerical values, were obtained from all the five data samples combined. After pre-processing the terms reduced to 4945 as shown in Table 1. After subsequently applying the BERT Encoder model the terms further reduced to 4257 as shown in the bottom right of Table 1. The datasets, Python codes and results of this study can be found in [25].

Table 1. Total number of words extracted after each stage, per sample S_i .

After	S_1	S_2	S_3	S_4	S_5	Total
Pre-processing	891	959	977	1145	973	4945
BERT Encoder	778	800	856	998	825	4257

The results in Table 1 show that the number of extracted terms reduced on all five samples (encoded $S_i, 1 \leq i \leq 5$) when passed through the BERT encoder block. The terms extracted by the BERT Encoder model became the input data that was fed into the WordNet for the identification of verbs and nouns.

4.3 Model Training Performance Results

We trained our model to label data based on the predefined categories. Realizing that multiple labels can exist in a single sentence we therefore treated our problem as a multi-label classification, and we used a multi-labeled sample dataset (baseline dataset) to train our model and we evaluated the performance of the model during training. A critical area of successfully training a good model

is to get the hyperparameters right. Pre-Trained Transformer models converge and produce good results if the hyperparameter values are reasonable. We conducted preliminary experiments to determine the optimal training parameters. The model was then fine-tuned using the Adam optimizer with the specified settings.

Table 2. Training Performance of the Model.

Epoch	Training loss	RMSE	F1-Score	ROC AUC	Accuracy
1	0.35	0.35	0.21	0.56	0.05
2	0.23	0.18	0.81	0.84	0.35
3	0.17	0.04	0.98	0.98	0.92
4	0.13	0.03	0.98	0.98	0.89
5	0.11	0.01	0.99	0.99	0.97
6	0.09	0.02	0.98	0.98	0.92

Table 2 shows the performance of the model during training. The training loss and the root mean square error (RMSE) reduced from 35% on the first epoch up to less than 10% on the sixth epoch. The model's low RMSE and low training loss indicate accurate predictions and minimal discrepancies between predictions and true labels, indicating good learning and close data fitting. The table shows that the accuracy of the model was initially very low at 5% during the first epoch but as we monitored the accuracy across the epochs the accuracy improved to 92% on the last epoch. The same applies to the F1-score and Receiver Operating Characteristic Area Under the Curve (ROC AUC) which also gradually improved to 98%.

The charts of accuracy, ROC AUC, F1-score and RSME are presented in Fig. 4. The curves in Fig. 4 show that as we increased number of epochs, the root mean square error decreased, whereas, the accuracy, ROC AUC and F1-score increased. We tested our model on five different data samples encoded S_1, S_2, S_3, S_4 , and S_5 , respectively. The performance of the model during testing are presented in Table 3.

Table 3. Testing Performance of the Model.

Data Sample	Accuracy	Precision	Recall	F1 Score
S_1	0.745	0.882	0.821	0.850
S_2	0.741	0.895	0.800	0.845
S_3	0.743	0.889	0.817	0.852
S_4	0.694	0.878	0.760	0.815
S_5	0.716	0.892	0.772	0.828

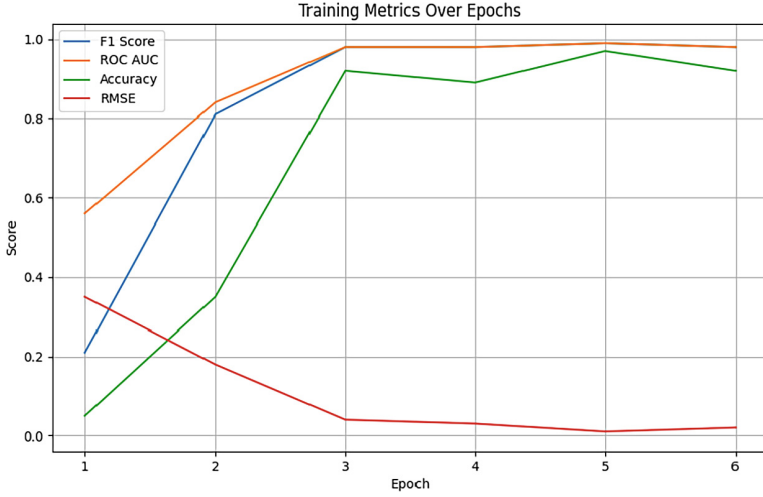


Fig. 4. Curves of Metrics Used to Evaluate the Model During Training.

From Table 3, we can see that the average F1-score across all five samples is 84%. Our model achieved good accuracy, exceeding 70% for all data samples and reaching 74.5% on the data sample S_1 .

Figure 5 present a chart showing the accuracy, precision, F1-score and Recall when testing the model on different data samples. The accuracy ranges between 69.4% to 74.5% for the five data samples tested while F1-score ranges between 81% and 85% indicating that the model was able to accurately extract category-specific terms from given data samples. When reviewing the testing performance across different five data samples $S_i, 1 \leq i \leq 5$, it is clear that the model's performance metrics i.e. accuracy, precision, recall and F1-score show variability depending on the specific data sample used. The variation can be attributed by differences in class distribution, data quality and feature representativeness across the samples. For example samples S_3 and S_2 have more balance between precision and recall. While data samples such as S_4 with potential noise and class imbalance exhibit lower accuracy and recall, suggesting the model struggles to generalize well on less representative or noiser samples.

Table 4 shows a sample processed terms that was input to the proposed model, the embeddings created by BERT and the correcting output verbs and nouns extracted from the initial text.

4.4 Comparison with Related Studies

The following studies [10–12], were chosen for comparison since they address a similar problem of term extraction from unstructured data and they all implement the BERT model as it is done in this study. The research studies used five self-produced datasets [10] through extraction of tweets from twitter and this is

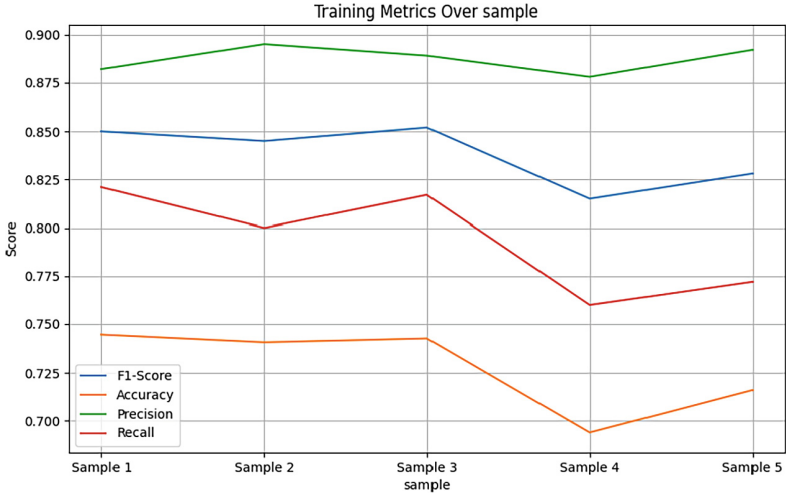


Fig. 5. Curves of Metrics Used to Evaluate the Model During Testing.

Table 4. Sample processed terms, embeddings, output verbs and nouns.

Terms	Embeddings	Verbs	Nouns
"harvest prior transplanting seedlings raised enclosed heated nurseries hardened increased"	[-0.01089761 -0.00806674 -0.00863286 -0.00839307 -0.00793375 -0.00847953 -0.00833057 -0.00831175 -0.00820751 -0.00825037]	["transplanting", "raised", "enclosed", "hardened", "increased"]	["harvest", "seedlings", "nurseries"]

similar to our work were we also used five self-produced datasets from extraction of horticulture terms from different FAO webpages. Lastly, these research studies use similar evaluation metrics such as Precision, Recall, F1 score and Accuracy, enabling easier comparison. The comparison should however be interpreted bearing in mind that different datasets were used.

The performance comparison results in Table 5 show that some related studies [11, 12] did not report their accuracy. The study [10] reported an accuracy of 83% that is higher than the accuracy of 73% achieved by our proposed model. This may be due to the differences in the datasets used and how these datasets were labeled. In this study, we used a multi-labeled dataset to cater for multiple categories that may exist in a single sentence. However, the results in Table 5 display that our proposed model achieved a precision of 89%, a recall of 79%, as well as an F1-Score of 84%, better than the related studies. This indicate the

effectiveness of our proposed model in correctly extracting terms from textual data.

Table 5. Comparison with other models.

Study	Accuracy	Precision	Recall	F1 score
Devika et al. (2021) [10]	83%	82%	78 %	83 %
Vaidhya et al. (2020) [11]	–	77%	73%	75%
Zhang et al. (2020) [12]	–	74%	72%	73%
Our Model	73 %	89 %	79 %	84 %

5 Conclusion and Future Work

In this paper, we presented an attention-based Deep Learning model for enriching term extraction from text. After pre-processing text obtained from web-pages, we employed the BERT encoder block to enrich the term extraction based on predefined categories. The contextualized embeddings were then checked for relatedness by comparing the cosine similarity scores of word embeddings with those of the category keywords. The embeddings with similarity scores above the predefined threshold were retained as the extracted text. After selecting domain-specific terms, they were passed to WordNet for knowledge context where the terms were labeled as either verbs or nouns. The proposed method incorporated contextualized term representation, domain information, knowledge context, and term disambiguation. The proposed model was tested on five data samples and evaluated with five metrics: accuracy, precision, recall, and F1-score. The experimental results show that the proposed model yields effective domain-specific term extraction and vocabulary labeling with an accuracy of 73%, a precision of 89%, a recall of 79%, as well as F1-Score of 84%, better than the related studies. Future work will explore experimenting with different pre-training techniques that can better capture syntactic and semantic relationships, enhancing the extraction of category-specific terms used in verb and noun prediction. Additionally, optimizing the BERT model for more precise and enriched term extraction presents another promising direction. Furthermore, integrating recent advances in multimodal large language models (LLMs) with visual representations offers an exciting opportunity to broaden the scope of research. This integration could pave the way for innovative applications across fields like term extraction, knowledge representation, and educational tools.

References

1. Tran, H.T.H., Martinc, M., Caporusso, J., Doucet, A., Pollak, S.: The recent advances in automatic term extraction: a survey. arXiv preprint [arXiv:2301.06767](https://arxiv.org/abs/2301.06767) (2023)

2. Navarro-Almanza, R., Juárez-Ramírez, R., Licea, G., Castro, J.R.: Automated ontology extraction from unstructured texts using deep learning. In: Castillo, O., Melin, P., Kacprzyk, J. (eds.) *Intuitionistic and Type-2 Fuzzy Logic Enhancements in Neural and Optimization Algorithms: Theory and Applications*. SCI, vol. 862, pp. 727–755. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-35445-9_50
3. Hegde, S., Garg, A., Murray-Rust, P., Mietchen, D.: Mining the literature for ethics statements: a step towards standardizing research ethics. *Res. Ideas Outcomes* **8**, e94685 (2022)
4. Si, Y., Wang, J., Xu, H., Roberts, K.: Enhancing clinical concept extraction with contextual embeddings. *J. Am. Med. Inf. Assoc.* **26**(11), 1297–1304 (2019)
5. Sarıtaş, K., Öz, C.A., Güngör, T.: A comprehensive analysis of static word embeddings for Turkish. *Expert Syst. Appl.* **252**, 124123 (2024)
6. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
7. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805)* (2018)
8. Liu, Y., et al.: Roberta: a robustly optimized bert pretraining approach. *arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692)* (2019)
9. Lu, W., Jiang, J., Shi, Y.: Application of entity-bert model based on neuroscience and brain-like cognition in electronic medical record entity recognition. *Front. Neurosci.* **17**, 1259652 (2023)
10. Devika, R., Vairavasundaram, S., Mahenthara, C., Varadarajan, V., Kotecha, K.: A deep learning model based on BERT and sentence transformer for semantic keyphrase extraction on big social data. *IEEE Access* **9**, 165252–165261 (2021)
11. Vaidhya, T., Kaushal, A.: Domain specific BERT representation for Named Entity Recognition of lab protocol. *arXiv preprint [arXiv:2012.11145](https://arxiv.org/abs/2012.11145)* (2020)
12. Zhang, R., et al.: Rapid adaptation of bert for information extraction on domain-specific business documents. *arXiv preprint [arXiv:2002.01861](https://arxiv.org/abs/2002.01861)* (2020)
13. Hättö, A.: Automatic term extraction for conventional and extended term definitions across domains (2020)
14. Frisoni, G., Moro, G., Carbonaro, A.: A survey on event extraction for natural language understanding: riding the biomedical literature wave. *IEEE Access* **9**, 160721–160757 (2021)
15. Fellbaum, C.: WordNet. *Theory and applications of ontology: computer applications*, pp. pp. 231–243. Springer, Dordrecht (2010)
16. Géron, A.: *Hands-on machine learning with scikit-learn. Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems*, vol. 1, (2022)
17. Panesar, S.S.: Biomedical knowledge discovery from unstructured text corpora using contextual word embeddings (2022)
18. Sharma, P., Li, Y.: Self-supervised contextual keyword and keyphrase retrieval with self-labelling (2019)
19. Yih, W.T., Goodman, J., Carvalho, V.R.: Finding advertising keywords on web pages. In: *Proceedings of the 15th International Conference on World Wide Web*, pp. 213–222, (2006)
20. Mihalcea, R. and Tarau, P.: TextRank: bringing order into text. In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 404–411 (2004)
21. Mitchell, R.: *Web Scraping with Python*. O'Reilly Media, Inc., Boston (2024)

22. Banerjee, S., Pedersen, T.: An adapted lesk algorithm for word sense disambiguation using wordnet. In: Gelbukh, A. (ed.) *CICLing 2002*. LNCS, vol. 2276, pp. 136–145. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45715-1_11
23. Wang, Y., Li, L., Wan, X., Wang, J.: Woven fabric defect detection based on the cascade classifier. In: *12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1–5. IEEE (2019)
24. Avola, D., et al.: A shape comparison reinforcement method based on feature extractors and f1-score. In: *International Conference on Systems, Man and Cybernetics (SMC)*, pp. 2155–2159. IEEE (2019)
25. Datasets and Python Codes. [Attention based term extraction : drive_link](#). Accessed 19 May 2024

Artificial Intelligence and Smart Systems



Temporal Analysis of Social Concerns on African Social Media: Insights from Topics, Themes, and Sentiments

Harriet Sibitenda¹✉, Awa Diattara¹, Assitan Traore²,
Elke Rundensteiner³, and Cheikh Ba¹

¹ Laboratoire d'Analyse Numérique et Informatique, Computer Science Department,
University of Gaston Berger, Saint-Louis, Senegal

{harriet.sibitenda, awa.diattara, cheikh2.ba}@ugb.edu.sn

² Grenoble, France

assitan.traore@free.fr

³ Data Science, Data-Driven Intelligence Systems Laboratory, Worcester Polytechnic
Institute, Worcester, MA, USA

rundenst@wpi.edu

Abstract. Understanding social media discussions about African development is crucial for policy-makers. This study analyses trends in these discussions by (1) reviewing data collection methods, (2) extracting key topics, themes, and sentiments, and (3) applying temporal analysis to observe trends and predict future trends. We compiled a dataset with 22,036 records from Twitter (X) and YouTube. Using BERTopic and Llama for topic extraction, we identified unique topics in six themes: poverty, hunger, education, employment, health, and security. Sentiment analysis was conducted with validation from human-annotated labels. By temporal analysis, we observed and predicted future trends for social concerns using Prophet model. We achieved a coherence score of 0.65 C-v for 304 topics, with 0.82 Kappa agreement. Llama2 outperformed BERTScore in theme extraction, with “Poverty” scoring an F1 of 0.89, followed by “Health” (0.76). Llama had a precision of 0.72 and balanced accuracy of 0.55 in sentiment analysis. Temporal analysis showed steady trends in poverty and security, with growing interest in health, education, and jobs. Positive sentiment peaked in 2020 around youth leadership, while governance and corruption remained stable. Findings inform policy decisions on Africa’s development, and future work will explore key social entities within posts.

Keywords: Social media analysis · Development issues · Temporal analysis · Large Language Models · Policy implications

1 Introduction

Motivation. Social issues, affecting large segments of society, arise when individual challenges become widespread [1]. These issues, including terrorism,

RSIF-PASET, UGB, WPI.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2025

Published by Springer Nature Switzerland AG 2025. All Rights Reserved

T. M. N. Ngatched et al. (Eds.): PAAISS 2024, LNCS 732, pp. 105–124, 2025.

https://doi.org/10.1007/978-3-031-94439-0_7

unemployment, and poverty [2], vary by cultural, economic, and political contexts. Africa's economic stagnation since 2019 stems from COVID-19 and the Ukraine conflict, which disrupted global stability [3]. Sociologists analyze these issues using surveys, experiments, observations, and existing datasets [4], providing insights for policymakers. With the rise of Social Network Platforms (SNPs), AI tools can analyze public sentiment and predict trends, aiding sustainable development [5]. SNPs like Facebook, Twitter, and Instagram generate vast user data through interactions and posts. Social feedback from these platforms is vital for social protection, as highlighted by the International Labor Organization (ILO) [6]. Monitoring social media helps policymakers address emerging concerns using demographic insights and strategic planning. Temporal analysis involves both observations and prediction of trends to examine how social concerns evolve by identifying patterns and behavioral shifts in social media data. It helps researchers track emerging trends and public sentiment over time, enabling policymakers to respond effectively. This analysis also reveals interactions between factors shaping societal responses to events. Time series forecasting predicts future social concerns by analyzing trends, seasonality, and noise in social media data [7]. A key step is testing for stationarity to ensure consistent statistical properties, crucial for many forecasting models [8]. Tools like the Test Statistic, p-value, and Critical Values assess stationarity and data dependencies [9–11]. Integrating this approach with social media analysis helps policymakers anticipate and respond to societal shifts.

Brief Related Literature. Recent research leverages time-series modeling for public health and economic challenges. Tian et al. [11] used a SARIMA model to forecast Hand, Foot, and Mouth Disease (HFMD) cases in China (2008–2018), capturing seasonal patterns but facing limitations like under-reporting and missing demographic data. Similarly, the power of time-series analysis to provide actionable insights into social phenomena is evident in the work of Oglesby-Neal et al. [12]. They used Twitter data to analyze shifts in public sentiment toward the police around the death of Freddie Gray in 2015. Their difference-in-differences analysis revealed a significant increase in negative sentiment, though the study struggled with the challenges of interpreting social media sentiment accurately.

The utility of time-series forecasting extends beyond public health and social sentiment to the economic sphere, as demonstrated by Papastefanopoulos et al. [13]. Their research compared traditional and modern time-series methods to forecast COVID-19 cases across ten countries. They found that traditional models like ARIMA and TBAT were generally more accurate, though the performance varied depending on specific factors across countries, with deep learning models underperforming due to limited data. Wu et al. [14], applied Transformer models to predict the incidence of Influenza-like Illness (ILI) based on historical time series data. By leveraging the self-attention mechanism, the Transformer model was able to capture complex temporal dependencies more effectively than traditional models like ARIMA and modern models like LSTM and Seq2Seq with attention.

Building on the exploration of economic forecasting, Menculini et al. [15] compared ARIMA, Prophet, and deep learning models for forecasting wholesale food prices. While deep learning models offered the most accurate predictions, they required significant computational resources. ARIMA models provided a competitive balance of accuracy and efficiency, despite challenges related to missing data.

In the realm of pandemic forecasting, Sah et al. [16] compared Prophet, ARIMA, and a hybrid LSTM-GRU model for forecasting COVID-19 cases in India. The hybrid model outperformed the others, especially in handling complex patterns, but the study emphasized the difficulties of forecasting in the dynamic context of a pandemic.

Together, these studies illustrate the diverse applications of time-series forecasting across different domains, demonstrating its critical role in public health, social analysis, economic forecasting, and beyond. Through the continued development and refinement of these models, researchers, and practitioners can enhance their ability to predict and respond to complex challenges in an increasingly data-driven world. In our study, we compared different models of time series forecasting on different findings. Our study integrates temporal analysis of observing trends and forecasting changes with common topics, top themes about development and sentiment scores of the social posts.

Objective. This research introduces AI-driven techniques to analyze social issues on African social media platforms, providing policymakers with actionable insights for sustainable development. Specifically, it aims to collect and analyze social media data to identify prevalent topics in key sectors such as healthcare, education, infrastructure, and security. By assessing the similarity of social comments to selected themes and exploring emotional correlations, the study examines public sentiment on various issues. Additionally, temporal analysis is applied to track trends over time, enabling the observation and prediction of future social concerns based on extracted topics, sentiments, and themes.

Our Approach. To analyze social concerns on African social media through temporal analysis, we propose a three-phase approach. First, we collect and explore social media data from platforms like Twitter and YouTube. Second, we apply topic modeling using BERTopic and Large Language Models (LLMs) like Llama2, extracting key themes and sentiment from comments with context-aware techniques such as Llama3. Finally, we compare temporal analysis models, including statistical methods like ARIMA and machine learning-based approaches like LSTM and Prophet, to track trends and forecast future concerns. Model performance is evaluated by examining relationships between topics, themes, and sentiments over time, providing insights into the evolving nature of social issues.

Our analysis of topic and sentiment extraction yielded significant insights. Using BERTopic and Llama, we identified 304 distinct topics with a coherence score of 0.65 C-v, indicating a well-balanced mix of diversity and clarity. The posts were grouped into six major themes: poverty, hunger, education, employment, health, and security, with a classification accuracy of 87%. In sentiment

analysis, FT-BERT stood out with an impressive F1 score of 0.998, while Llama performed well in extracting contextual sentiments, achieving a precision of 0.72 and a balanced accuracy of 0.55. For temporal analysis, the Transformer emerged as the top model, with a weighted score of 0.50 for themes and 0.84 for topics, excelling in normalized MSE and RMSE. The analysis revealed persistent trends across all categories, with poverty-related topics showing consistent engagement, particularly in neutral sentiment. Themes like “corruption and governance” and “youth empowerment” maintained steady attention, while positive sentiment peaked in 2020, especially around youth leadership, before declining. Negative sentiment remained low and stable. Forecasts suggest rising interest in themes like health, education, and employment, while security and poverty are expected to remain steady. The key contributions derived from our methodology are as follows:

- We enhance topic modeling by combining LLMs like Llama with BERTopic for text clustering, improving interpretability and relevance through concise, meaningful short-sentence labels.
- Using LLMs like Llama, we automated relevance filtering to extract theme titles on African development (health, education, security, poverty) and assessed document similarity to enhance thematic analysis.
- We use unsupervised models (TextBlob, BERT-nlptown, VADER) and LLMs (Llama) to classify social media sentiment (positive, negative, neutral), linking public emotions to development themes.
- We compared traditional, deep learning, and LLM-based methods for sentiment analysis and topic extraction, validating AI-generated labels against human annotations to ensure accuracy.
- To identify the most effective methods for predicting social concerns from African social media data, we explored trends and applied both statistical and machine learning models for time-series forecasting, focusing on assessing model accuracy for forecasting.
- We provided policymakers with a comprehensive understanding of evolving social discussions on sustainable development within African social media platforms, aiding in informed decision-making.

After the introductory section, a summary of the rest of the study includes the following sections: Sect. 2 covers the collection and exploration of social concerns based on topics, themes, and sentiments, as well as the methods applied for time series analysis and model evaluation. Section 3 presents the outcomes of temporal analysis, including trend observations and predictions, evaluation of models, and the integration of findings for topics, themes, and sentiments over time. Section 3.2 discusses our findings, comparisons to related studies, and limitations of the study. Section 4 outlines the study’s conclusions and provides recommendations for future work.

2 Methodology

Our study follows a structured approach to sentiment, topic, and theme analysis based on the Knowledge Discovery in Databases (KDD) framework by Fayyad et al. [17]. (1) The process begins with the collection of social media posts from platforms like Twitter and YouTube, focusing on discussions related to development. (2) Data preparation is performed, including rigorous cleaning, normalization, and transformation to prepare the text for analysis. (3) Various methods for extracting topics, sentiments, and themes are explored, including lexicon-based approaches, neural models, and LLMs like Llama3, to identify the most effective techniques for comprehensive analysis. (4) Temporal analysis tracks changes in trends for topics, themes, and sentiments, utilizing statistical methods such as ARIMA and machine learning models like LSTM, with model performances evaluated by analyzing trends and correlations over time. (5) Finally, insights from the analysis are integrated to offer a comprehensive view of public sentiment, topic dynamics, and social concerns, providing policymakers actionable information for sustainable development. Figure 1 displays a summary of the steps we followed to analyze social concerns from social posts based on the KDD process.

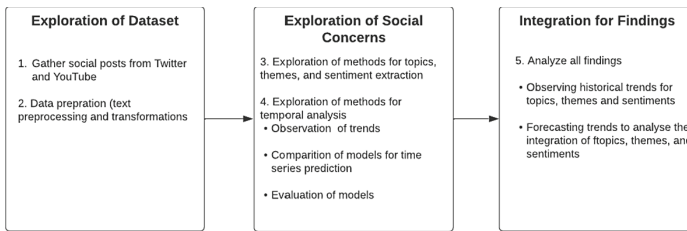


Fig. 1. Architecture for Time Series Analysis using the KDD process of knowledge discovery adopted from Fayyad et al., 1996

2.1 Dataset Description

In accordance with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA), we analyzed studies on collecting social posts. We defined sampling parameters for Facebook, Twitter, and YouTube and used the “Publish or Perish” tool to collect papers from 2018 to 2022, screening 2,180 articles and identifying 298 relevant ones. Data collection utilized five methods: self-reported surveys, public APIs, existing databases, manual tracking, and web scraping. Automated techniques, mainly APIs and web scraping, were used to gather data from Twitter and YouTube, focusing on persistent social issues in Africa from 2018 to 2023. Although Facebook is more popular, Twitter and YouTube were chosen for easier data access.

Our study focuses on African social issues, shaped by the 17 global sustainable development determinants [18]. We identified ongoing issues from

2018 to 2023 using keywords like “social problem” and “social challenge.” To reflect Africa’s diverse social media landscape, we extracted 18,853 records from YouTube (via Selenium and BeautifulSoup) and 9,199 from Twitter (via SnsCrape). The dataset includes YouTube titles, descriptions, and Twitter texts (under 200 words), predominantly in English (29,078 records), with smaller contributions in French, German, Spanish, and Italian. This approach provides insights into social issues within the African context.

Preparation of Dataset. We applied text preprocessing techniques including data integration, cleaning, and normalization. Files were combined, schemas established, and additional columns added. Text was aggregated, duplicates removed, and stored in MySQL. The cleaning process corrected errors, handled emojis, filled missing values, translated non-English content to English, and standardized the dataset. Normalization involved tokenizing text, removing stop words, and lemmatizing. The final dataset, SMA, contained 22,036 English records, converted into numerical form for machine learning. Text data was transformed into numerical features using statistical methods and word embeddings. Traditional techniques like One-hot encoding, Bag-of-Words, and TF-IDF were used but faced issues with data sparsity and high dimensionality [19]. To address this, word embeddings like Word2Vec, GloVe, and FastText captured semantic relationships more effectively. Dynamic embeddings (ELMO, BERT, GPT) were explored, with BERT chosen for its balance of contextual understanding and efficiency. For feature selection, PCA, T-SNE, and UMAP were evaluated, with UMAP selected for its speed, compact clustering, and low computational cost [20].

2.2 Exploration of Social Concerns from Social Comments

We aimed to identify distinct categories of discussion related to key sectors of development, along with the associated sentiments, in social comments. To achieve this, we explored common discussion topics and sentiments linked to sustainable development by reviewing existing NLP techniques, such as topic modeling, document similarity, and sentiment extraction, to uncover key themes and sentiments from social comments and link them to important development sectors.

Topic Extraction. We explored various methods for topic discovery, selecting BERTopic for its strong coherence and interpretability [21, 22]. BERTopic uses a class-based TF-IDF approach to identify relevant token words, enhancing topic specificity and clustering. It also supports dynamic modeling to track topic evolution over time. To refine topic labels, Llama2 converted token words into concise, descriptive labels and generated theme titles focused on African development, improving interpretability for policy analysis. We evaluated topic quality using coherence (C-v) and divergence (IRBO) metrics to measure semantic consistency and differentiation, crucial for reliable insights into social concerns. Coherence (C-v) gauges word similarity within topics, while divergence (IRBO) ensures distinct topic separation. For classification, we used F1 score, accuracy, precision,

and recall to validate model performance in categorizing social media content, providing a balanced view of topic relevance and classification accuracy, making results accessible and reliable for users.

Theme Extraction. We evaluated methods for classifying documents by sustainable development themes, selecting cosine similarity via BERTScore for its ability to assess semantic closeness between document vectors. This method captures contextual similarities, essential for identifying thematic overlaps. Additionally, we used a custom Llama2 prompt to extract themes aligned with content, benefiting from LLMs’ contextual comprehension. An 80% similarity threshold was set for applying multi-labels across six themes: hunger, poverty, security, health, employment, and education. To assess the model’s effectiveness, we used a multi-label classification strategy with OneVsRestClassifier and evaluated performance using precision, F1 score, recall, and accuracy. These metrics provide a comprehensive view of the model’s ability to accurately and relevantly assign themes, ensuring it captures the interconnected nature of sustainable development issues.

Sentiment Extraction. We used a combination of sentiment analysis techniques, including lexicon-based methods (like; SentiWordNet), corpus-based approaches (like; TextBlob, VADER), and advanced models like BERT-nlptown and Llama3. Lexicon-based methods provide basic polarity, while corpus-based approaches add contextual nuance. Llama3 and fine-tuned models like BERT and RoBERTa captured complex emotional tones, improving sentiment classification accuracy, especially for subtle emotions in social discourse. We assessed model performance using accuracy, precision, recall, and F1 score, metrics well-suited for sentiment analysis as they measure both correctness and consistency. These metrics ensure that the model reliably detects nuanced sentiment across various social concerns, providing a balanced evaluation of model effectiveness in capturing the range of emotions expressed in social media content. This approach guarantees that our sentiment analysis remains accurate and contextually relevant, offering insights into the emotional dimensions of social issues.

Metrics for Assessing Human Annotation Consistency. We used Inter-Annotator Agreement (IAA) metrics to evaluate annotation reliability. Krippendorff’s Alpha handled multiple annotators and missing data, while Pearson correlation measured the linear relationship between annotations. We also used Percentage Agreement to assess consensus. These metrics helped determine annotation consistency.

2.3 Existing Methods of Temporal Analysis

Temporal analysis helps uncover patterns and trends over time by structuring data in key steps: preparation, resampling, and visualization. Data preparation begins with converting the Date column into date-time format, and the Date column is set as the index of the data frame. By indexing with date values, the data becomes primed for efficient grouping and aggregation at various intervals,

laying a foundation for identifying temporal patterns. After converting dates to a usable format and setting them as the index, data is resampled into daily, weekly, monthly, and yearly intervals to capture different time-based patterns. Aggregating posts by specific columns of interest such as topics, themes, sentiment at each interval reveals shifts in public opinion. Finally, visualization across these intervals highlights specific trends, providing a comprehensive view of how social concerns and post volume change over time. This approach enables data-driven insights that support timely decision-making.

Methods for Time Series Forecasting. Statistical models like Moving Averages (MA), Exponential Smoothing (ES), and Holt-Winters methods are commonly used for time series forecasting. MA methods, including Simple Moving Average (SMA) and Weighted Moving Average (WMA), smooth short-term fluctuations but may struggle with seasonality, as they assume equal weight across time periods or use simple patterns [23,24]. Exponential Smoothing (ES), which assigns higher weights to recent data, is more responsive to recent changes and works well for data with seasonality but struggles with trends. Simple Exponential Smoothing (SES) works best for data without trends or seasonality [24]. Holt's Exponential Smoothing extends ES by adding a trend component, while Holt-Winters further incorporates seasonality, making it ideal for series with trends and seasonal patterns. However, Holt-Winters can be less accurate than more advanced methods like ARIMA and may be sensitive to outliers and large datasets.

Autoregressive Moving Averages (ARMA) combine autoregressive (AR) and moving average (MA) models to capture trends and patterns in time series data. ARIMA (AutoRegressive Integrated Moving Average) extends ARMA by including differencing to make data stationary [25]. ARIMA uses three parameters: p (autoregressive terms), d (differences for stationarity), and q (lagged forecast errors). It is a flexible model suitable for various time series, with variations like SARIMA (Seasonal ARIMA) and AUTO-ARIMA for specific forecasting needs. However, ARIMA assumes linear relationships, limiting its effectiveness for non-linear patterns or complex seasonality [26,27].

Hybrid Statistical Approaches like TBAT combine multiple models: Trigonometric seasonal formulation, Box-Cox transformation, ARMA errors, and a trend component. This makes TBAT effective for complex seasonal time series, improving decomposition and forecasting accuracy [13]. It is computationally efficient for maximum likelihood estimations and handles nonlinear features common in real-world data. However, TBAT requires careful tuning of various hyperparameters, which can be time-consuming and resource-intensive, limiting its practicality for quick solutions.

Prophet, developed by Facebook, is a generative additive model designed for forecasting time series data, focusing on trends, seasonality, and holidays, making it ideal for business applications [15,28]. It decomposes time series into trend, seasonality, and holiday components, allowing predictions over various timeframes. Prophet is robust against outliers, missing data, and sudden trend shifts, with an easy interface that supports adjustments. However, it requires

data in a specific format, adding complexity, and may struggle with accuracy in scenarios requiring high precision [16].

Machine Learning-Based Methods: Traditional machine learning algorithms are used to identify linear relationships in univariate time series data. Common algorithms include Bayesian Neural Networks (BNN), Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Radial Basis Functions (RBF), and Gaussian Processes [29,30]. These methods use historical data patterns for future predictions. For example, linear regression models relationships between variables, while LASSO regression aids in feature selection and regularization. SVM handles regression problems effectively, especially with non-linear data, by mapping inputs to higher-dimensional spaces. However, these methods struggle with multivariate datasets, where interdependencies affect performance.

Deep learning has greatly improved time series forecasting, especially for multivariate data. Algorithms like Multilayer Perceptrons (MLP), Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Deep Neural Networks (DNN), and Transformers enhance prediction accuracy by capturing complex patterns and dependencies [14,31]. Tools such as LSTNet, GP-Copula, Nbeats, and DeepAR further extend deep learning’s capabilities. However, these models require significant computational resources, leading to slower performance and may struggle with irregular events like holidays [28].

In our study we compared models of ARIMA, Prophet, LSTM, DeepAR and Transformer for prediction. Table 1 displays a summary of existing methods and their limitations.

Table 1. Comparison of Forecasting Methods

Method & Description	Limitations
Statistical Models: ES, SES, Holt-Winters	Assumes constant trend/seasonality, struggles with non-stationary data and sudden changes, sensitive to outliers
Autoregressive Moving Averages (ARIMA, SARIMA):	Requires stationary data, struggles with non-linear relationships, sensitive to parameter selection
Hybrid Statistical Approaches (TBAT):	Hyperparameter tuning is resource-intensive, may not generalize well to new patterns
Prophet:	Requires predefined seasonal effects, performs poorly on volatile or irregular data
Traditional ML Algorithms (SVM):	Struggles with temporal dependencies, requires extensive feature engineering
Deep Learning Algorithms (LSTM, RNN):	Computationally expensive, prone to overfitting, struggles with irregular patterns

For time series prediction, we first run the test of stationarity to data. We employed the Augmented Dickey-Fuller (ADF) test to assess the stationarity of time series data for sentiment, topics, and themes. The ADF test is crucial for determining whether a time series has constant statistical properties, such as mean and variance, over time. Stationarity is a key assumption for many

forecasting models, as non-stationary data can lead to inaccurate predictions. The test examines the presence of a unit root in the data, which would indicate non-stationarity. Based on the test results, we could decide whether to apply transformations like differencing to make the time series stationary before proceeding with predictive modeling. The test results were analyzed by comparing the test statistic to critical values at different significance levels (1%, 5%, 10%) and evaluating the p-value to determine stationarity.

Then we fixed the data from the initial four years to predict the trends for the fifth year, 2023. This process allowed us to assess how well the models captured historical trends and patterns to make near-term forecasts. We used models such as Prophet, ARIMA, LSTM, DeepAR, and Transformer to predict the final year's data based on the first four years. Once the predictions for the fifth year were generated, we extended the forecast further into the future using the best-performing model, selected based on its weighted score. This model was then employed to project long-term trends, providing a robust forecasting mechanism for themes, sentiments, and topics. This approach helped ensure the accuracy of short-term predictions and allowed us to evaluate model performance in the context of time-series data.

2.4 Evaluation Metrics for Time Series

Evaluation metrics for time series models are essential for assessing prediction accuracy. Common metrics include Mean Absolute Error (MAE), which measures average error size without considering direction; Root Mean Squared Error (RMSE), which penalizes larger deviations more heavily and is sensitive to outliers; Mean Absolute Percentage Error (MAPE), which expresses errors as percentages of actual values but can be misleading for small actual values; and R-squared (R^2), which indicates how well the model explains the variance in the data [32,33]. These metrics together provide a comprehensive view of forecast accuracy and model performance.

3 Results

3.1 Results for Extraction of Social Concerns

In our study, we identified the best-performing models for extracting topics, themes, and sentiments, along with their annotator agreement for ground truth assessment (see Sect. 2.2). For topic modeling, the combination of BERTopic and Llama achieved the highest performance, with a topic coherence score (C-v) of 0.81 and a topic divergence (IRBO) of 0.58, extracting 304 distinct topics. The annotators' agreement for topics resulted in a Kappa correlation of 0.82 and a balanced accuracy of 0.95. In theme analysis, the OneVsRest classification with ensemble voting emerged as the leading model, with "Poverty" achieving the highest F1 score of 0.89, followed by "Health" at 0.76, "Employment" at 0.73, "Education" 0.71, "Security" at 0.54, and "Hunger" at 0.52. Themes by Llama2 slightly outperformed BERTScore with a user percentage agreement of 0.30 and

a similarity score of 0.62. For sentiment analysis, FT-BERT outperformed other models with an impressive F1 score of 0.988, while Llama demonstrated strong contextual sentiment capture, achieving a precision of 0.72 and a balanced accuracy of 0.55. Annotators' percentage agreement for sentiments, derived through Llama3 prompt, reached 0.44, with a Pearson's correlation at 0.54. Overall, the results provided valuable insights into public attitudes toward key development issues in the African social media dataset.

Observing Yearly Trends with Social Concerns. Figure 2 shows the trends in posts for various topics over daily, weekly, monthly, and yearly intervals. Daily and weekly trends exhibit fluctuations, with spikes in topics like “SNL reactions” and “spiritualist travels” around 2020. From 2021 onwards, topics such as “corruption and governance” and “youth empowerment” show gradual growth. Monthly data indicates sustained engagement, with topics like “water & environmental challenge” maintaining steady activity. Yearly data highlights long-term trends, with “youth empowerment” and “governance” consistently growing, while “spiritualist travels” peaked in 2020 and then declined. In general, the yearly data reveals broader trends of rising interest in governance, youth empowerment, and environmental issues.

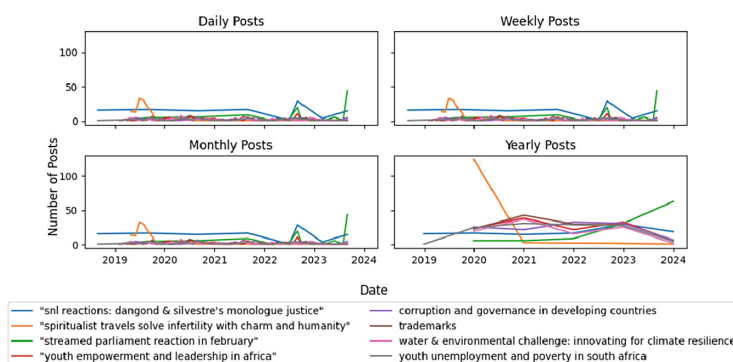


Fig. 2. Shows a sample of trends for Topics

The Fig. 3 shows distinct patterns in different time frames. On a daily basis, positive sentiment exhibits sharp peaks, especially towards the end of each year, with notable increases in 2024. Negative sentiment is relatively low in daily observations, showing minimal spikes and a gradual decrease over time. On a weekly scale, positive sentiment also dominates with regular spikes, while negative sentiment remains low with smaller peaks. Neutral sentiment remains largely stable, with only minor fluctuations. The monthly trends reveal a recurring pattern for positive sentiment, particularly at the end of each year, while negative sentiment continues its downward trend with less pronounced variations. Neutral sentiment displays slight periodic changes, but remains relatively flat. Across

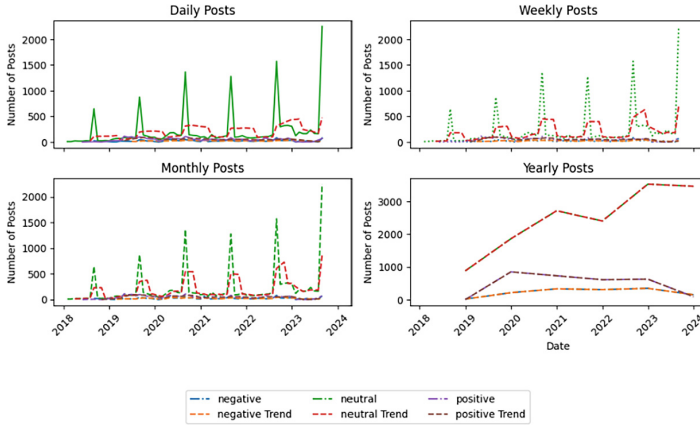


Fig. 3. Shows a sample of trends for Sentiments

all time frames, the data consistently show a shift towards more positive sentiment, with negative sentiment gradually decreasing and neutral sentiment holding steady. Overall, the data suggest a shift towards more positive sentiment over time, with negative sentiment decreasing and neutral sentiment holding steady.

Figure 4 shows the trends in posts across themes like hunger, poverty, education, health, employment, and security. Daily and weekly posts show frequent spikes, likely tied to specific events or media cycles. Monthly data smooths these fluctuations, with poverty and health showing the most consistent engagement. Yearly trends reveal steady growth from 2018 to a peak in 2021, influenced by the pandemic's impact on health, employment, and poverty. Post volumes decrease slightly after 2021, with poverty remaining the most discussed theme, followed by health and employment, while hunger and security see the least engagement. These insights suggest that while short-term engagement is driven by specific events, larger social trends sustain discussions on poverty, health, and employment over time.

Forecasting Trends with Social Concerns. We forecast trends of social concerns based on the existing dataset, see Sect. 2.3. The Dickey-Fuller test results show that the sentiment time series (183 records) is non-stationary, with a test statistic of -1.186979 and a high p-value of 0.679172 , indicating that transformation is needed for better model performance. In contrast, the topics (2,726 records) and themes (1,382 records) time series are stationary, with test statistics of -4.812250 and -3.664678 , and p-values of 0.000052 and 0.004638 , respectively. These results suggest that topics and themes are more suitable for prediction models like ARIMA without further transformation.

We train our models on the first four years (2018 to 2022) and then predict trends for 2023. We select the best model based on evaluation metrics and then make future predictions of 2024 and more, see Sect. 2.4. Since we have actual

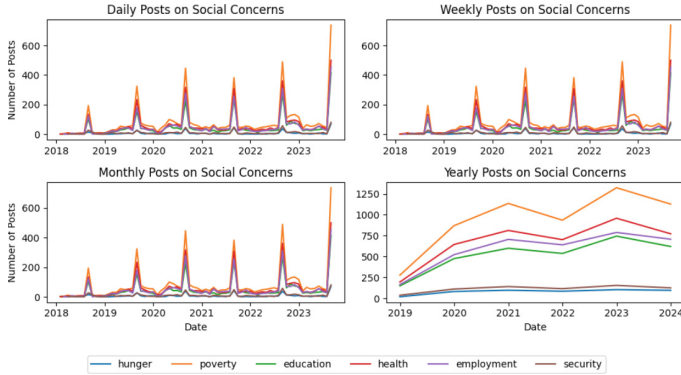


Fig. 4. Shows a sample of trends for Themes

trends, we then evaluate the performance of models for the existing test data in 2023.

Table 2 shows the performance of various models in predicting themes and topics. For themes, Transformer achieved the highest weighted score (0.49527), outperforming ARIMA (0.49190) and DeepAR (0.48859), though ARIMA had the lowest error metrics. For topics, Transformer had the highest score (0.84188), slightly outperforming Prophet (0.84000), indicating its strength in handling time series data across multiple categories.

Overall, Transformer proves to be the top model for themes, while Prophet also shines in topic forecasting, with ARIMA and DeepAR offering competitive results across categories. For further forecasting, we selected Prophet model to predict future trends in topics because of its added advantage to predict holiday trends. We studied the integration of future topics, based on the themes and sentiment.

3.2 Discussions of Trends for Integrated Findings

Analysis of Trends on Topics and Sentiments. Figure 5 shows trends in posts on various topics, categorized by sentiment (Neutral, Negative, Positive) from 2018 to 2023. Most discussions are neutral, with topics like “overcoming anxiety disorder,” “hacking for business,” and “streamed parliament reaction in February” growing, especially in 2023 and 2024. Negative sentiment topics, such as “corruption and governance” and “spiritual infertility consultation,” had fewer posts and minimal fluctuations. Positive sentiment, peaking in 2020 for topics like “youth empowerment” and “spiritualist travels,” has declined but shows a resurgence in areas like “water & environmental challenge.” Overall, neutral sentiment dominates, with positive sentiment declining and negative sentiment staying low.

Forecasting trends in topics and their sentiment reveals key insights into future discussions. In neutral sentiment, topics like overcoming anxiety disorder

Table 2. Weighted Scores and Metrics for Models (Themes and Topics)

Model	MA. ↓	RMS. ↓	MAP. ↓	R ² ↓	Norm. MSE ↓	Norm. MAE ↓	Norm. RMSE ↓	Norm. MAPE ↓	Weight. score ↑
<i>Themes</i>									
Prophet	10.92	32.33	199.87	−0.05	0.016	0.043	0.106	0.398	0.444
ARIMA	2.93	6.31	92.97	−0.05	0.000	0.000	0.000	0.030	<u>0.492</u>
LSTM	4.19	7.52	152.32	−0.49	0.000	0.007	0.005	0.235	0.336
DeepAR	2.86	6.52	89.99	−0.07	0.000	0.000	0.001	0.020	0.489
Transformer	13.16	34.11	93.18	−0.14	0.018	0.055	0.113	0.031	0.495
<i>Topics</i>									
Prophet	136.63	252.41	107.09	−0.23	1.000	0.709	1.000	0.079	<u>0.840</u>
ARIMA	99.64	169.79	84.22	−0.01	0.452	0.513	0.664	0.000	0.756
LSTM	106.65	169.49	99.21	−0.01	0.451	0.550	0.663	0.052	0.752
DeepAR	191.56	240.99	242.39	−0.12	0.911	1.000	0.954	0.545	0.814
Transformer	146.95	237.70	139.45	−0.09	0.887	0.764	0.940	0.190	0.842

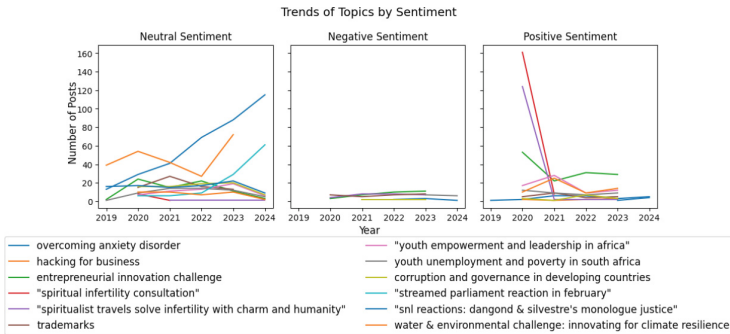


Fig. 5. Observation of posts about various topics, categorized by sentiment

and entrepreneurial innovation show stability with slight fluctuations, expected to maintain steady interest. In negative sentiment, themes such as corruption, governance, and youth unemployment display flat trends, indicating no significant rise in negative discussions. Positive sentiment topics, like youth empowerment and SNL reactions, show temporary peaks but are expected to taper off by 2025. Water and environmental challenges exhibit fluctuating interest, indicating continued but varied engagement. Overall, most themes will persist with stable discussions and consistent engagement.

Analysis Between Themes, Sentiment, and Topics. Figure 6 examines how topics related to poverty intersect with sentiment from 2018 to 2023. Neutral sentiment shows the most consistent activity, with topics like “corruption and governance,” “youth empowerment,” and “spiritualist travels” receiving steady attention, especially from 2021 to 2023. Negative sentiment shows minimal engagement, with few posts and limited spikes in topics like “hip hop instrumentals” and “corruption and governance.” Positive sentiment peaked in 2020, especially with “spiritualist travels,” but declined sharply afterward. Overall, poverty dis-

cussions are mostly neutral, with less engagement in both negative and positive sentiments.

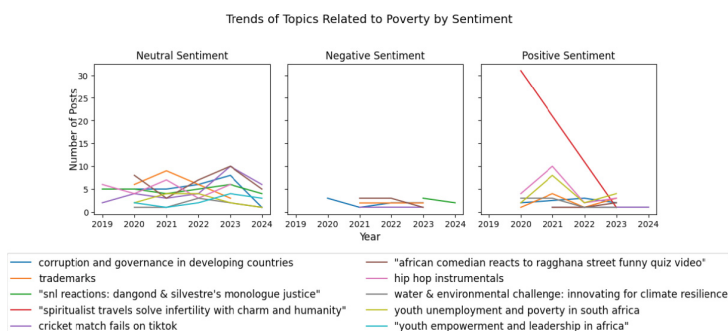


Fig. 6. Shows a sample of correlation of sentiments and themes

Forecasting trends in themes using Prophet reveals distinct patterns. Hunger and health show recurring spikes, especially around 2020 and 2022, with moderate fluctuations ahead. Education and employment peak sharply in 2020 and 2023, indicating continued discussions, while security shows steady, lower peaks, and stable future interest. Poverty follows a consistent trend with few fluctuations. Overall, health, education, and employment are expected to see increased attention, while security and poverty remain stable.

Figure 7 shows predicted trends for poverty-related topics and their sentiments. Under neutral sentiment, “trademarks” shows volatility, while “corruption and governance” and “cricket match fails on TikTok” remain stable. Negative sentiment topics, like “hip hop instrumentals” and “cricket match fails on TikTok,” show flat trends with low engagement. In positive sentiment, topics like “spiritualist travels” and “youth empowerment” display sharp fluctuations, while “hip hop instrumentals” and “water & environmental challenge” remain stable with moderate interest.

3.3 Summary of Overall Findings

To summarize, our analysis of development-related discussions on social media revealed key insights. Using BERTopic and Llama for topic extraction, we identified 304 unique topics, achieving a coherence score of 0.65 C-v, indicating a solid balance between topic diversity and clarity. Annotator agreement on topics reached a Kappa correlation of 0.82 with a balanced accuracy of 0.95. Posts were categorized into six key themes: poverty, hunger, education, employment, health, and security, with a classification for “Poverty” achieving the highest F1 score of 0.89, followed by “Health” at 0.76, “Employment” at 0.73. Themes by Llama2 achieved a marginally higher performance than BERTScore, with a user agreement rate of 30% and a similarity rating of 0.62. For sentiment analysis,

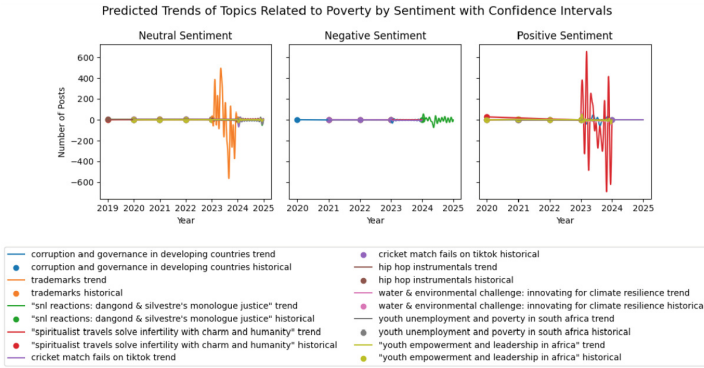


Fig. 7. Shows a sample of correlation of sentiments and themes

FT-BERT achieved an excellent F1 score of 0.998, while Llama proved to be the best model for extracting contextual sentiments, with a precision of 0.72 and balanced accuracy of 0.55. We used human-annotated labels to validate the models, resulting in moderate agreement, with Krippendorff’s Alpha at 0.42 and an overall percent agreement of 0.53. These results highlight the strong synergy between AI techniques and human input for analyzing emotional tones and social concerns, especially in the African context.

Furthermore, our study explored trends and predictions in social media discussions across various topics, themes, and sentiments related to development issues. We analyzed how these topics evolved from 2018 to 2023, identifying persistent and declining trends. Topics related to poverty, for example, showed consistent engagement under neutral sentiment, particularly in themes like “corruption and governance” and “youth empowerment.” Positive sentiments peaked in 2020, while negative sentiment remained low throughout the years. The use of advanced forecasting techniques revealed that themes like health, education, and employment are likely to experience higher future interest, while security and poverty are expected to maintain stable but consistent engagement. Overall, our methods highlighted both short-term spikes and long-term stability, providing a comprehensive understanding of how social concerns evolve in public discourse.

Findings from Related Studies. Tian et al. [11] used a SARIMA model to forecast hand, foot, and mouth disease (HFMD) cases in China from 2008 to 2018. The SARIMA (1, 1, 2) (0, 1, 1)₁₂ model captured the seasonal patterns of disease incidence with high accuracy. The model achieved a mean error rate (MER) of 16.86% and an R^2 of 94.27%, indicating strong predictive performance for capturing periodic and seasonal variations in the dataset. The study by Wu et al. [14] demonstrated that Transformer models outperformed ARIMA and LSTM for forecasting Influenza-like Illness (ILI), achieving RMSE of 0.593. However, the study focused on health-related data with clear seasonality, whereas in our study, Transformer performance on social issues like poverty and security may be limited by less predictable patterns and data scarcity. Sah et al.

[16] used Prophet, ARIMA, and a hybrid Stacked LSTM-GRU model to forecast COVID-19 cases in India. The study found that the hybrid LSTM-GRU model outperformed the other models at 0.74 RMSE and 0.699 R2. Prophet and ARIMA, while effective, were less accurate than the hybrid deep learning model just like in our performances.

In our study, we applied the Transformer model to forecast trends in social media topics related to development issues. The Transformer model provided robust predictions, particularly for themes like health, education, and employment, with a weighted mean score of 0.49 for themes and 0.52 for sentiments and 0.84 for top ten topics. Our use of time-series forecasting demonstrated consistent performance across multiple categories, showing the model’s capacity to capture both short-term spikes and long-term trends.

Limitations of Our Study. In our study, while time-series forecasting models like Prophet offered useful insights, they rely on historical data, which may not account for sudden shifts in social discussions. The models assume past trends will continue, potentially missing changes due to major events. Sparse data for certain topics also limited prediction accuracy, particularly for less-discussed themes. Additionally, focusing on yearly trends may overlook finer, short-term fluctuations. LLMs like Llama for extraction of social concerns require significant computing resources, such as the A100 GPU, which may not be accessible to all, including governments or nonprofits, during this major shift in AI technology.

Our study has successfully uncovered key insights into the evolution of social concerns on social media, using advanced methods for topic extraction and sentiment analysis. In the future, we plan to extend this work by a deeper exploration of social entities within the discussions to better understand their influence and interconnectedness within the social media network. This approach will provide more nuanced insights into how specific individuals, groups, or topics impact public sentiment and discourse. Additionally, we aim to develop methods for tracking how these entities drive conversations and influence shifts in social concerns, offering valuable foresight into emerging issues in global development.

4 Conclusion

In conclusion, our research used time series forecasting to analyze trends in African social media discussions on development issues. We aimed to: (1) establish methods for collecting and analyzing social media data, creating the Social Media in Africa (SMA) dataset with 22,036 records from Twitter (X) and YouTube; (2) extract topics and sentiments, identifying key themes like poverty, hunger, education, employment, health, and security; and (3) use temporal analysis to observe trends and predict the evolution of these topics, themes, and sentiments.

Our findings from topic and sentiment extraction were noteworthy. Using BERTopic and Llama for topic extraction, we identified 304 unique topics, achieving a coherence score of 0.65 C-v, showing a strong balance between topic diversity and clarity. Posts were categorized into six key themes: poverty, hunger,

education, employment, health, and security, with a classification accuracy for “Poverty” achieving the highest F1 score of 0.89, followed by “Health” at 0.76, “Employment” at 0.73. For sentiment analysis, FT-BERT excelled with an F1 score of 0.998, while Llama demonstrated strong performance in contextual sentiment extraction, with a precision of 0.72 and balanced accuracy of 0.55. For temporal analysis, we observed persistent trends across all categories. Topics related to poverty showed consistent engagement, particularly in neutral sentiment, with themes like “corruption and governance” and “youth empowerment” maintaining steady attention. Positive sentiment peaked in 2020, especially around themes like youth leadership, but has since declined. Negative sentiment remained low and stable throughout the study period. Forecasting by Prophet revealed that themes such as health, education, and employment are likely to see increased future interest, while security and poverty will maintain stable engagement. Transformer proved the most effective model for forecasting, achieving a weighted score of 0.50 for themes and 0.84 for topics, excelling in normalized MSE and RMSE metrics, but to predict holiday seasons we selected the Prophet model.

In summary, our study highlights the value of combining topic and sentiment extraction with time series forecasting to track and predict the trajectory of public discourse on development issues. Moving forward, we will explore how social entities influence public sentiment and discourse, providing deeper insights into the dynamics of global development conversations.

Acknowledgement. We would like to express our gratitude to all those who have contributed to the completion of this research project. Firstly, we are thankful to the PASET-RSIF, for their financial support, without which this research would not have been possible.

References

1. Pawluch, D.: Social problems. *The Social Science Encyclopedia*, pp. 949–951 (2004). https://doi.org/10.5005/jp/books/14198_13
2. Obileye, A.A., Aborisade, R.: Chapter Twenty Social Problem. *Introduction to Sociology: African Culture, Context and Complexity*, pp. 276–285 (2020)
3. Zeufack, A.G., et al.: Africa’s Pulse, No. 25, vol. 25 (2022). <https://doi.org/10.1596/978-1-4648-1871-4>
4. Martin, B.: *Sociological Research Methods*, pp. 36–67 (2017). <https://doi.org/10.4324/9781315129952>
5. Hartmann, J., Huppertz, J., Schamp, C., Heitmann, M.: Comparing automated text classification methods. *Int. J. Res. Mark.* **36**(1), 20–38 (2019). <https://doi.org/10.1016/j.ijresmar.2018.09.009>
6. ILO: *Africa Regional Social Protection Strategy, 2021-2025* (2021)
7. Tablatin, C.L.S., Acosta, M.E., Patacsil, F.F.: Analyzing community needs in the city of Urdaneta using text analytics. *WSEAS Trans. Inf. Sci. Appl.* **20**, 170–177 (2023). <https://doi.org/10.37394/23209.2023.20.20>

8. Rahimi, I., Chen, F., Gandomi, A.H.: A review on COVID-19 forecasting models. *Neural Comput. Appl.* **35**(33), 23671–23681 (2021). <https://doi.org/10.1007/s00521-020-05626-8>
9. Büyüksahin, Ü.Ç., Ertekin, Ş.: Improving forecasting accuracy of time series data using a new ARIMA-ANN hybrid method and empirical mode decomposition. *Neurocomputing* **361**, 151–163 (2019). <https://doi.org/10.1016/j.neucom.2019.05.099>
10. Garcia, D., Rimé, B.: Collective emotions and social resilience in the digital traces after a terrorist attack. *Psychol. Sci.* **30**(4), 617–628 (2019). <https://doi.org/10.1177/0956797619831964>
11. Tian, C.W., Wang, H., Luo, X.M.: Time-series modelling and forecasting of hand, foot, and mouth disease cases in China from 2008 to 2018. *Epidemiol. Infect.* **147**, 1–3 (2019). <https://doi.org/10.1017/S095026881800362X>
12. Oglesby-Neal, A., Tiry, E., Kim, K.: Public Perceptions of Police on Social Media: A Big-Data Approach to Understanding Public Sentiment toward the Police, pp. 1–12 (2019)
13. Papastefanopoulos, V., Linardatos, P., Kotsiantis, S.: COVID-19: a comparison of time series methods to forecast percentage of active cases per population. *Appl. Sci. (Switzerland)* **10**(11), 1–15 (2020). <https://doi.org/10.3390/app10113880>
14. Wu, N., Green, B., Ben, X., O'Banion, S.: Deep transformer models for time series forecasting: the influenza prevalence case, pp. 1–10 (2020). <http://arxiv.org/abs/2001.08317>
15. Menculini, L., et al.: Comparing prophet and deep learning to ARIMA in forecasting wholesale food prices. *Forecasting* **3**(3), 644–662 (2021). <https://doi.org/10.3390/forecast3030040>
16. Sah, S., Surendiran, B., Dhanalakshmi, R., Mohanty, S.N., Alenezi, F., Polat, K.: Forecasting COVID-19 pandemic using prophet, ARIMA, and hybrid stacked LSTM-GRU models in India. *Comput. Math. Methods Med.* **2022**, 1–19 (2022). <https://doi.org/10.1155/2022/1556025>
17. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: From data mining to knowledge discovery in databases. *AI Mag.* **17**(3), 37–53 (1996)
18. Banerjee, B.: Sustainable Development Goals. DK Taneja's Health Policies and Programmes in India, p. 28 (2017). <https://doi.org/10.5005/jp/books/13071.5>
19. Gautam, A.K., Bansal, A.: Performance analysis of supervised machine learning techniques for cyberstalking detection in social media. *J. Theor. Appl. Inf. Technol.* **100**(2), 449–461 (2022)
20. Kobak, D., Linderman, G.C.: Initialization is critical for preserving global data structure in both t-SNE and UMAP. *Nat. Biotechnol.* **39**(2), 156–157 (2021). <https://doi.org/10.1038/s41587-020-00809-z>
21. Sibitenda, H., et al.: Extracting semantic topics about development in Africa from social media. *IEEE Access* **12**, 142343–142359 (2024). <https://doi.org/10.1109/ACCESS.2024.3466834>
22. Grootendorst, M.: BERTopic: neural topic modeling with a class-based TF-IDF procedure, pp. 1–10 (2022). <http://arxiv.org/abs/2203.05794>
23. Chung, H., Shin, K.S.: Genetic algorithm-optimized long short-term memory network for stock market prediction. *Sustainability (Switzerland)* **10**(10), 1–18 (2018). <https://doi.org/10.3390/su10103765>
24. Parmezan, A., Souza, V., Batista, G.: Evaluation of statistical and machine learning models for time series prediction: identifying the state-of-the-art and the best conditions for the use of each model. *Inf. Sci.* **484**, 302–337 (2019). <https://doi.org/10.1016/j.ins.2019.01.076>

25. Siامي-Namini, S., Namin, A.S.: Forecasting economics and financial time series: ARIMA vs. LSTM, pp. 1–19 (2018). <http://arxiv.org/abs/1803.06386>
26. Chae, S., Kwon, S., Lee, D.: Predicting infectious disease using deep learning and big data. *Int. J. Environ. Res. Public Health* **15**(8), 1–20 (2018). <https://doi.org/10.3390/ijerph15081596>
27. Fathi, O.: Time series forecasting using a hybrid ARIMA and LSTM model. Velvet Consulting, pp. 1–7 (2019)
28. Alazab, M., Awajan, A., Mesleh, A., Abraham, A., Jatana, V., Alhyari, S.: COVID-19 prediction and detection using deep learning. *Int. J. Comput. Inf. Syst. Ind. Manage. Appl.* **12**, 168–181 (2020)
29. Singh, S., Yassine, A.: Big data mining of energy time series for behavioral analytics and energy consumption forecasting. *Energies* **11**(2), 1–26 (2018). <https://doi.org/10.3390/en11020452>
30. Ardabili, S.F., et al.: COVID-19 outbreak prediction with machine learning. *Algorithms* **13**(10), 1–36 (2020). <https://doi.org/10.3390/a13100249>
31. Giuliani, F., Hasan, I., Cristani, M., Galasso, F.: Transformer networks for trajectory forecasting. In: *Proceedings - International Conference on Pattern Recognition*, pp. 10335–10342 (2020). <https://doi.org/10.1109/ICPR48806.2021.9412190>
32. Jain, A., Sukhdeve, T., Gadia, H., Sahu, S.P., Verma, S.: COVID19 prediction using time series analysis. In: *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*, pp. 1599–1606 (2021). <https://doi.org/10.1109/ICAIS50930.2021.9395877>
33. Li, W., Law, K.: Deep learning models for time series forecasting: a review. *IEEE Access* **12**, 92306–92327 (2024). <https://doi.org/10.1109/ACCESS.2024.3422528>



Real-Time Object Detection Using an Ensemble of One Stage and Two Stage Object Detection Models with Dynamic Fine-Tuning Using Kullback-Leibler Divergence

Nkalubo Lenard Byenkya¹ , Nakibuule Rose², and Okila Nixon³

¹ Department of Data Science and Artificial Intelligence, Kyambogo University, Kampala, Uganda

lenkalubo@gmail.com

² Department of Computer Science, Makerere University, Kampala, Uganda
rnakibuule@cit.ac.ug

³ Department of Computer Science, Lira University, Lira, Uganda

Abstract. This paper proposes an ensemble approach to real-time object detection that combines the strengths of YOLOv5 and Faster R-CNN. YOLOv5 is used to generate a set of object proposals, then Faster R-CNN is used to refine these proposals and produce more accurate object detection results. To further improve the accuracy of our object detection results, a cascade refinement network that uses dynamic fine-tuning is utilized, which uses the Kullback-Leibler divergence to dynamically adjust the weights of the Faster R-CNN model based on the confidence scores of the YOLOv5 object proposals. Our approach is evaluated on a novel dataset collected in Uganda with other State-of-the-art approaches which include RetinaNet, Cascade R-CNN, Single-Shot MultiBox Detector, and R-CNN. Experimental results revealed that our proposed ensemble model outperforms both base models with an average precision of 0.96, which is significantly higher than that of YOLOv5 (0.91) and Faster R-CNN (0.90). Our model was also able to achieve real-time inference speeds, processing frames at a rate of 25 frames per second, the same speed achieved by YOLOv5, faster than the speed of 15 frames per second by Faster R-CNN. Our results also revealed that the proposed ensemble model is comparable to other state-of-the-art object detection models. Our proposed approach can be used to improve the accuracy and speed of real-time object detection in a variety of applications.

Keywords: Faster R-CNN · Kullback-Leibler divergence · Object detection · YOLOv5

1 Introduction

One of the core tasks in computer vision is object detection, which seeks to identify and categorize things in a picture or video [1]. In many applications, such as robotics, surveillance systems, and autonomous driving, it is essential [2]. Significant progress

has been achieved in object detection methods over time, resulting in the creation of multiple effective models [3].

There are two main methods in the field for object detection: one-stage and two-stage models [4]. Single Shot MultiBox Detector (SSD) and You Only Look Once (YOLO) are examples of one-stage models that estimate item bounding boxes and class probabilities directly in a single run. Applications requiring quick and effective object identification can benefit from these models' well-known real-time capabilities [3, 4]. However, two-stage models use a two-step procedure, including Region-based Convolutional Neural Networks (Faster R-CNN) and Mask R-CNN [2]. Region proposals are created in the first step, and they are then categorized and improved in the second step. Although two-stage models are more accurate, they frequently take longer and need more computing power [5].

In order to capitalize on the advantages of many models and enhance overall performance, ensemble approaches have been extensively investigated in the object identification field [6]. By combining the outputs of multiple models, ensemble methods can achieve better accuracy and robustness. However, in dynamic environments where objects, backgrounds, and lighting conditions may change, static ensemble models may not be able to adapt effectively [7]. To address this limitation, dynamic fine-tuning techniques have emerged as a promising approach. Dynamic fine-tuning involves continuously updating the ensemble models using new data or adapting them based on environmental changes. This allows the models to adapt and improve their performance over time.

Kullback-Leibler (KL) divergence is a widely used method for measuring the difference between two probability distributions [8]. It provides a measure of how one distribution diverges from another. By using KL divergence in the dynamic fine-tuning process, the ensemble models can learn from new data and adjust their parameters to better match the current distribution of objects and backgrounds in the environment.

In this study, we propose an ensemble of one-stage and two-stage object detection models with dynamic fine-tuning using Kullback-Leibler divergence. We hypothesize that the proposed model presents an exciting opportunity to achieve real-time object detection with improved accuracy and adaptability in dynamic environments such as blind movement guidance systems. This study aims to explore this approach and evaluate its performance through extensive experiments and analysis. This study is aimed at developing an ensemble model for real-time object detection that combines the strengths of two state-of-the-art object detection algorithms. YOLOv5 is a fast object detection model that is well-suited for real-time applications. It can detect objects at a speed of up to 45 frames per second. However, YOLOv5 is not as accurate as some other object detection models, such as Faster R-CNN. Faster R-CNN is a more accurate object detection model, but it is not as fast as YOLOv5. It can detect objects at a speed of up to 25 frames per second. By combining YOLOv5 and Faster R-CNN, the ensemble model can achieve the speed of YOLOv5 and the accuracy of Faster R-CNN. This makes it a good choice for real-time applications that require high accuracy. The novelty of the work of developing an ensemble using YOLOv5 and Faster R-CNN is that it combines the strengths of two different object detection models to achieve better performance than either model could achieve on its own.

1.1 Motivation and Objectives

Real-time object detection is a critical task in computer vision with applications in various domains. While significant progress has been made in object detection algorithms, there are still challenges to address, such as achieving high accuracy and adaptability in dynamic environments. One-stage and two-stage object detection models have their respective strengths and weaknesses. One-stage models provide real-time performance but may sacrifice accuracy, while two-stage models offer higher accuracy but require more computational resources. The motivation behind this study is to leverage the strengths of both approaches by developing an ensemble of one-stage and two-stage models. Dynamic fine-tuning is a technique that allows models to adapt and improve their performance over time. By continuously updating the ensemble models using new data or adapting them to changing environmental conditions, the models can better handle variations in object appearance, background, and lighting conditions. This dynamic adaptation is particularly important in real-time scenarios where the environment is constantly changing. The use of Kullback-Leibler divergence in the dynamic fine-tuning process adds a novel aspect to the research. KL divergence enables the models to measure the difference between the current distribution of objects and the distribution learned from the initial training data. This allows the models to dynamically adjust their parameters and improve their accuracy by aligning with the current data distribution. The motivation for this study is to explore the potential benefits of combining an ensemble of one-stage and two-stage object detection models with dynamic fine-tuning using Kullback-Leibler divergence. By leveraging the strengths of both approaches and adapting the ensemble models in real time, it is anticipated that this research can lead to improved accuracy and adaptability in real-world object detection scenarios. The outcomes of this study can have significant practical implications, such as enhancing the performance of object detection systems such as those used in the guidance of blind movements, enabling more robust surveillance systems, and improving object detection capabilities in robotics applications. The objective of this work is to propose an ensemble of one-stage and two-stage object detection models, with and without dynamic fine-tuning using Kullback-Leibler divergence, to evaluate their individual and combined contributions to real-time object detection accuracy and adaptability in dynamic environments. The proposed approach aims to demonstrate that integrating dynamic fine-tuning enhances the ensemble's performance, while a comparative analysis establishes the necessity and impact of each component.

The main research objectives for the study are:

- i. To propose a new method for real-time object detection that combines one-stage and two-stage object detection models.
- ii. To introduce a new technique for dynamic fine-tuning of object detection models.
- iii. To evaluate the proposed method on a variety of object detection datasets.

1.2 The Paper Structure

The rest of the paper is organized as follows. Section 2 discussed the Related Work, Sect. 2.1 discussed Object Detection Models, Sect. 2.2 discussed a Premier on object detection models, Sect. 2.3 discussed Transfer Learning in Object Detection, Sect. 2.4

discussed Dynamic Fine-tuning and Sect. 2.5 discussed Kullback-Leibler Divergence in Transfer Learning; Sect. 3 discussed Methodology, Sect. 3.1 discussed Dataset Description, Sect. 3.2 discussed the Proposed Algorithm and Sect. 3.3 discussed Evaluation Metrics; Sect. 4 discussed Experimental Results, Sect. 4.1 discussed Comparative Performance of the Proposed Ensemble Model with Base Models, Sect. 4.2 discussed Effect of Dynamic Fine-tuning on Object Detection Accuracy and Sect. 4.3 discussed Comparison with State-of-the-art Approaches; Sect. 5 discussed the results and Sect. 5.1 discussed Computational Complexity of the Proposed Model and Sect. 6 discussed Conclusion and Future Work.

2 Related Work

2.1 Object Detection Models

The computer vision problem of object detection is locating and identifying items in pictures or films [9]. Applications for object detection models are numerous and include robots, security systems, and self-driving automobiles. Conventional detectors and deep learning-based detectors are the two categories into which the object detection framework separates detectors. The two-stage detector and the one-stage detector are the two types of Deep-learning object detectors. Unified detectors [10–14] when performing classification and regression without first creating a sparse region of interest (RoI) collection, employ dense anchor boxes, whereas in two-stage detectors [2, 15–17], Sparse region proposals are developed in the first stage of two-stage detectors, following which they are regressed and classified [18]. One-stage detectors are more effective due to their straightforward architecture, although two-stage detectors still exceed them in terms of accuracy. The accuracy gap still exists despite recent attempts to enhance one-stage detectors by copying the structural design of two-stage detectors [18].

On the other hand, single-stage detectors treat object detection as a simple regression problem that accepts the entire image as input and simultaneously generates multiple bounding boxes and class probabilities. The model is hence considerably quicker than the two-stage object detectors [19]. Two-stage detectors provide adequate accuracy, but their computation times are considerable. Therefore, one-stage detectors are advised to process in less time while maintaining adequate accuracy [20]. While RCNN, Fast RCNN, and Faster RCNN algorithms are instances of two-stage detector algorithms, SSD and YOLO with their variants are examples of one-stage model algorithms. According to the study [18], two-stage detectors are superior to one-stage ones in the following ways: 1) One-stage detectors have a class imbalance issue if no additional design is added because they immediately face every region on the image. On the other hand, two-stage detectors eliminate the majority of the unfavorable recommendations by selecting a small number of region proposals. 2) The head of the network (used for proposal classification and regression) can be larger with two-stage detectors because they examine fewer proposals than one-stage detectors do. This makes it possible to extract richer features. 3) Two-stage detectors employ the RoI Align operation to extract the location-consistent feature from each sampled proposal, as opposed to one-stage detectors that might permit different region proposals to share the same feature and could lead to severe feature misalignment because of the coarse and spatially implicit representation of the proposals.

4) In comparison to single-stage methods, two-stage detectors do a double regression of the object location (once on each step). The fundamental issue that all one-stage detectors face, the mismatch between anchor boxes and convolutional features, significantly reduces one-stage detector performance [10]. One-stage object detectors' inability to outperform top-performing, two-stage algorithms is primarily due to class imbalance, claims the study [11]. The study suggested the focused loss, which adjusts the cross-entropy loss and concentrates learning on difficult negative examples, as a remedy. The creation of a fully convolutional one-stage detector and detailed experimental analysis proving that it reaches cutting-edge accuracy and speed were used to demonstrate the study's effectiveness.

2.2 A Premier on Object Detection Models

a) YOLO (You Only Look Once) and Its Variants

The YOLO (You Only Look Once) [1] family of object identification algorithms is renowned for its accuracy and quickness. As single-shot detectors, YOLO models make predictions about the bounding boxes and class probabilities of every object in an image in a single pass. Compared to region-based object detection models like Faster R-CNN, which first create a set of region proposals before classifying each one as either containing or not containing an object, YOLO models are therefore far faster. Convolutional neural networks (CNNs) were employed in the initial YOLO model, YOLOv1, a single-stage model that predicted bounding boxes and class probabilities for objects in an image. YOLOv1 [21] achieved state-of-the-art accuracy on the PASCAL VOC object identification dataset and real-time rates (45 frames per second). Numerous variations of the YOLO model have been proposed since YOLOv1. YOLOv2 [21] added a deeper CNN and a more sophisticated loss function to YOLOv1 to increase its accuracy. YOLOv3 [4] substantially enhanced YOLOv2's accuracy by employing a more potent CNN and a more effective bounding box representation. YOLOv4 [22] and YOLOv5 [23] have further enhanced the speed and accuracy of YOLO models. Self-driving cars, video surveillance, and medical imaging are just a few of the many applications that already make extensive use of YOLO models. Additionally, YOLO models are utilized in mobile applications like image editing and augmented reality. Nevertheless, two-stage object detectors like Faster R-CNN are more accurate than YOLO models [24]. This is because two-stage detectors first create a set of candidate bounding boxes and then refine them, whereas YOLO just predicts bounding boxes and class probabilities in a single run.

b) Single Shot MultiBox Detector

The Single Shot MultiBox Detector (SSD) [3] is an object detection technique that predicts bounding boxes and class probabilities for items in an image using a single neural network. It predicts the bounding boxes and class probabilities for every object in an image in a single pass because SSD is a one-stage detector. When compared to area-based object identification models like Faster R-CNN, which first create a set of region proposals before classifying each one as either containing or not containing an object, SSD is significantly faster. In order to function, SSD splits the image into a grid of cells. A set of bounding boxes and class probabilities are predicted for each cell based on the potential locations of objects. A regression layer is used to predict the

bounding boxes, while a classification layer is used to forecast the class probabilities. The ImageNet dataset, a massive image collection with more than 1 million photos and 1,000 item classifications, served as the training set for SSD. SSD has been utilized in a number of applications, such as medical imaging, video surveillance, and self-driving automobiles, and it was able to attain state-of-the-art accuracy on the ImageNet dataset. However, two-stage object detectors like Faster R-CNN are more accurate than SSD.

c) Region-Based Convolutional Neural Networks (R-CNN)

A set of object detection techniques known as region-based convolutional neural networks (R-CNN) [25] locate items in an image by using region recommendations. Girshick and colleagues' publication "R-CNN: Fast, Accurate, Region-based Object Detection with Convolutional Neural Networks" (2014) was the first to introduce R-CNN. Using an image, R-CNN initially creates a collection of region recommendations. The classification of these region proposals as either object-containing or object-non-containing follows. The object's location is then approximated within each proposed region. Given that R-CNN is a two-stage detector, it first creates a collection of area proposals before classifying each one as either object-containing or not. Because of this, R-CNN is more accurate but slower than single-stage object detection models like YOLO. The PASCAL VOC object detection dataset was used to train R-CNN. More than 10,000 photos with more than 20 item categories are included in PASCAL VOC. On the PASCAL VOC dataset, R-CNN achieved state-of-the-art accuracy. Video surveillance and self-driving cars are only two of the many uses for R-CNN. The disadvantage of RCNN is that it requires three different models to detect targets and processes several areas using CNN, which increases prediction time. Fast-RCNN employs a laborious and time-consuming technique.

d) Faster R-CNN

Faster R-CNN [2] is an object detection approach that shares features between the object detection network (ODN) and the region proposal network (RPN), increasing the speed and accuracy of R-CNN. In order for Faster R-CNN to function, the RPN must first produce a set of region proposals from an image. The ODN then categorizes these region ideas as either carrying or not carrying an object. After then, an estimate of the object's position is made for each region proposal. Given that Faster R-CNN is a two-stage detector, it first creates a collection of region proposals before classifying each one as either object-containing or not.

e) Cascade R-CNN

An expansion of the well-liked Faster R-CNN object detection framework is Cascade R-CNN [26]. By employing a series of steps to progressively improve the detection outcomes, it seeks to increase object detection's accuracy and resilience. Several detector stages are trained in a cascaded fashion in Cascade R-CNN, with each step concentrating on enhancing detection quality at varying degrees of difficulty. In order to gradually lower false positives and increase localization accuracy, the output of one stage is used as the input for the subsequent step. When compared to single-stage or two-stage detectors, cascade R-CNN has demonstrated better accuracy and resilience. It works especially well in difficult situations where there is a lot of background clutter, small item sizes, or a high

degree of object occlusion. In contrast to single-stage detectors like YOLO, Cascade R-CNN necessitates several steps of training and inference, increasing the computational complexity [27]. The detection model is run several times at each step of the cascade, which lengthens the inference time.

f) RetinaNet

As a single-stage model, RetinaNet [28] makes predictions about bounding boxes and class probabilities for every object in an image in a single pass. This makes RetinaNet faster than region-based object detection models, like Faster R-CNN, which classify each region proposal as either containing or not containing an object after first generating a collection of region proposals. To extract features from an image, RetinaNet employs a feature pyramid network (FPN). A number of convolutional layers organized in a pyramid-like structure make up the FPN. The FPN extract's upper layers feature at a finer level, whilst its lower layers feature at a coarser level. As a result, RetinaNet can identify objects at different sizes. RetinaNet uses a single convolutional neural network (CNN) to forecast bounding boxes and class probabilities for objects in an image. A set of bounding boxes and class probabilities are output by the CNN after receiving an image as input. A regression layer is used to predict the bounding boxes, while a classification layer is used to forecast the class probabilities. A large-scale object detection dataset called COCO was used to train RetinaNet. More than 200,000 photos with more than 80 object types are included in COCO. RetinaNet was able to attain state-of-the-art accuracy on the COCO dataset. Fast, accurate, and simple to train, RetinaNet is a potent object detection model. RetinaNet has been used in a range of applications, including self-driving cars, video surveillance, and medical imaging. However, in order to recognize objects, RetinaNet uses anchor boxes of different scales and aspect ratios. Compared to models that employ more sophisticated bounding box regression approaches, this approach may produce less accurate localization even while it offers reasonable coverage of various object sizes and shapes [29].

2.3 Transfer Learning in Object Detection

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task [30]. This can be done by freezing the weights of the model trained on the first task and only training the last few layers on the second task. This can be a very effective way to train object detection models, as it can significantly reduce the amount of training data required. According to [30], transfer learning (TL) is characterized in terms of task and domain. A feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$ make up the domain D , where $X = \{x_1, \dots, x_n\} \in \mathcal{X}$. Given a particular domain, $\mathcal{D} \in \{\mathcal{X}, P(X)\}$, a task consists of two parts: a label space $\mathcal{Y}$ and an objective prediction function $f(\cdot)$, which is learned from the training data consisting of pairings, which consist of pairs $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in \mathcal{Y}$. Predicting the appropriate label, $f(x)$, of a new instance x is possible using the function $f(x)$. Transfer learning seeks to enhance the learning of the target prediction function $f_T(\cdot)$ in the target domain $\mathcal{D}_T$ utilizing the knowledge in the source domain $\mathcal{D}_s$ and the learning task $\mathcal{T}_s$, where $\mathcal{D}_s \neq \mathcal{D}_T$, or $\mathcal{T}_s \neq \mathcal{T}_T$. There are two main approaches to transfer learning for object detection: (1) Feature extraction: In this approach, a pre-trained model, such as a

VGGNet or ResNet, is used as the feature extractor for a region-based object detection algorithm, such as Faster R-CNN or YOLOv3. In this approach, the weights of the pre-trained model are frozen, and only the last few layers of the network are trained on the object detection task. This can significantly reduce the amount of training data required, as the pre-trained model has already learned to extract features from images that are relevant to object detection. (2) Fine-tuning: In this approach, a pre-trained model is used as the backbone for a fully convolutional network (FCN). In this approach, the weights of the pre-trained model are fine-tuned on the object detection task. This can be done by using a loss function that combines the loss from the classification task and the loss from the object detection task. This approach can be more effective than the first approach, as it allows the model to learn to associate features with object classes.

2.4 Dynamic Fine-Tuning

A machine learning method called dynamic fine-tuning enables a model's parameters to be continuously updated as fresh data becomes available [31]. In contrast, typical machine learning techniques use a fixed dataset to train the model, which is subsequently deployed without any changes. Because a model may learn from new data and adjust to changes in the underlying distribution of the data, dynamic fine-tuning can be utilized to gradually enhance a model's performance. This can be especially helpful when the data is dynamic or when the model is being used in a setting where the data is continuously changing. Dynamic fine-tuning can be implemented in a variety of ways. One popular strategy is to employ a method known as online learning. Online learning is a subset of machine learning in which the model is trained using newly available data. This implies that new information is continuously added to the model, which over time may serve to enhance its performance. A method known as transfer learning is another way to dynamic fine-tuning. In machine learning, transfer learning is the process of using a model that has been trained on one task as a foundation for training a model on another. When not enough data is provided to train a model from scratch, this can be helpful. A potent method for gradually enhancing machine learning models' performance, increasing their adaptability, and cutting down on training time is dynamic fine-tuning. However, because the model must be updated each time new data becomes available, dynamic fine-tuning can also be computationally costly [32]. For dynamic fine-tuning in this work, we employed the transfer learning approach. Equation 1 below illustrates the dynamic fine-tuning method.

$$w = w - \eta * \nabla w l(x, y) \quad (1)$$

where: w is the model's parameters, η is the learning rate, $\nabla w l(x, y)$ is the gradient of the loss function with respect to w , x is a data point from the target domain and y is the label for x . The error between the model's predictions and the ground truth labels is quantified using the loss function. One hyperparameter that regulates the speed at which the model's parameters are changed is the learning rate. Until the loss function converges or a maximum number of iterations is reached, the dynamic fine-tuning algorithm is repeated.

2.5 Kullback-Leibler Divergence in Transfer Learning

The degree to which one probability distribution differs from another is measured by Kullback-Leibler Divergence (KLDs) [33]. The difference between the entropy functions of the two distributions is its definition. The uncertainty associated with a probability distribution is measured by its entropy function. A machine learning technique called transfer learning uses a model that has been learned on one task as a foundation for training a model on another [34]. In circumstances where there is insufficient data to train a model from scratch, this can be helpful. One method for enhancing transfer learning models' performance is dynamic fine-tuning with KL divergence [35]. KLD can be used to quantify how comparable the source and target distributions are in transfer learning [33]. The distribution of the data used to train the pre-trained model is known as the source distribution. The distribution of the data being used to refine the model is known as the target distribution. The degree of similarity between the source and target distributions increases with decreasing KLD. Depending on how each layer contributes to the model's loss, it dynamically chooses which layers to fine-tune. The similarity between the weights of the pre-trained model and the fine-tuned model is measured using the KL divergence. The most probable layers to gain from fine-tuning are those with the biggest KL divergence. It has been demonstrated that dynamic fine-tuning with KL divergence enhances the performance of transfer learning models on a range of tasks, such as natural language processing, picture classification, and object recognition. This indicates that there is a greater chance that the pre-trained model will perform well on the intended data set. Equation 2 defines KLD in the context of transfer learning.

$$KL(P||Q) = \sum_x P(x) * \log\left(\frac{P(x)}{Q(x)}\right) \quad (2)$$

where P is the source distribution, Q is the target distribution and x is a variable in the feature space. The KL divergence measures the amount of information that is lost when P is assumed to be Q . This means that it calculates the separation between the two distributions. In transfer learning, the distribution of the data used to train the model is usually the source distribution. This distribution is the distribution of the data to which the model is being applied.

3 Methodology

3.1 Dataset Description

In this study, we used datasets collected around Kampala, Uganda. The image data were collected in the morning, afternoon, and evening, during both dry and rainy environments, both indoor and outdoor, and since we plan to deploy our model to our local setting to help people with blind impairments in their movements, the collected images were from an environment where blind people move, and interact with their peers and families around Kyambogo University. We collected 1000 RGB images with 20 classes (pothole_road, dish, dog, donkey, door, fence, flowers, grass, person, pothole, sign_post, stairs, stones, tarmac, trees, bicycle, Boda, building, car, and walk_way). The images were collected

using an ordinary smartphone with the specifications: Infinix Hot 40i (RAM: 8 + 8GB, ROM: 256GB), 6.6" HD + Display, 50MP Dual AI Rear Camera, 32MP Crystal-Clear Front Camera, 5000 mAh Battery With 18W Fast Charge. We applied preprocessing and argumentation techniques before the dataset was split into training, testing, and validation. During data processing, we resized the dataset into 640x640 which is the standard size for yolov5, after which we applied auto orientation. For data augmentation, the following operations were performed: Horizontal flip, rotation of 90 degrees both clockwise and counter-clockwise, rotation between -15 degrees and 15 degrees, shear of + -10 horizontal and vertical, grayscale to 15% of the images, Hue between -15 degrees and 15 degrees, saturation between -25 degrees and 25 degrees, brightness between -15 degrees and 15 degrees, exposure between -10 degrees and 10 degrees, blur of up to 2.5px and noise of up to 0.1% of pixels. The data was then split as the following: 70% for training, 20% for validation, and 10% for testing. Figure 1 shows some of the images we collected.



Fig. 1. The sample of the Image dataset used during this study.

3.2 The Proposed Algorithm

In this section, we describe the proposed Ensemble algorithm shown in Fig. 2. The proposed model is a combination of one-stage and two-stage object detection models. Under one-stage object detection, YOLOv5 while Faster R-CNN was considered for two-stage object detection. The main objective of this study, we propose an ensemble approach to real-time object detection that combines the strengths of two state-of-the-art object detection algorithms. Both YOLOv5 and Faster R-CNN were selected because they are very fast object detection models. YOLOv5 can detect objects at a speed of up to 45 frames per second, while Faster R-CNN can detect objects at a speed of up to 25 frames per

second. They are fast, accurate, flexible, and easy to use. This makes them well-suited for real-time applications. The model starts by having an image as input, and then YOLOv5 is used to quickly generate a set of object proposals. Then, Faster R-CNN is used to refine the object proposals generated by YOLOv5. The weights of the Faster R-CNN model are dynamically fine-tuned using Kullback-Leibler divergence. Kullback-Leibler divergence is a measure of the difference between two probability distributions. In this case, the two probability distributions are the predicted probability distribution of object classes from YOLOv5 and the ground truth probability distribution of object classes. The weights of the Faster R-CNN model are updated so that the predicted probability distribution more closely matches the ground truth probability distribution. The output of the algorithm is a set of object detection results, which includes the bounding boxes, class labels, and confidence scores of the objects detected in the image. The parameters for the proposed ensemble model include the following: 1) The parameters of the YOLOv5 model; Which includes the parameters of the backbone network, the head network, and the loss function; and 2) The parameters of the Faster R-CNN model: This includes the parameters of the backbone network, the region proposal network, the classification network, and the regression network. Therefore, the parameters of the ensemble method include the parameters from the two models. The Fig. 2 shows the pseudocode of the model.

- 1 **Input:** Model parameters
- 2 **Output:** Object Detection
- 3 Start:
- 4 *Load and preprocess dataset*
- 5 *Load pretrained models*
- 6 Build an ensemble Model:
 - a) *Take an input image.*
 - b) *Pass the image through each model*
 - c) *Calculate the KL divergence between the predictions of each model and the ground truth annotations.*
 - d) *Calculate the weight of each model based on the KL divergence.*
 - e) *Combine the weighted predictions from all models to obtain the ensemble prediction.*
 - f) *Apply non-maximum suppression (NMS) to remove overlapping bounding boxes.*
 - g) *Evaluate the ensemble predictions using the evaluation metrics.*
- 7 Implement dynamic fine-tuning:
 - a) *Monitor the performance of each model on a validation set during inference.*
 - b) *Identify the models with lower performance based on the evaluation metrics.*
 - c) *Retrain the identified models using the training dataset.*
 - d) *Update the ensemble with the newly trained models.*
 - e) *Repeat steps 6b to 6g with the updated ensemble.*
- 8 Repeat step 7 until the desired performance is achieved.

Fig. 2. The Pseudocode of the proposed Model

3.3 Evaluation Metrics

The proposed model was evaluated using Average precision (AP): AP is a measure of the overall accuracy of an object detection model. It is calculated by averaging the precision scores at different recall levels. Mean average precision (mAP): mAP is the average of AP scores over a set of object categories. It is a more comprehensive measure of accuracy than AP, as it takes into account the performance of the model on all object categories. Accuracy: Accuracy is a measure of the overall performance of the model. It is calculated by dividing the number of correct predictions by the total number of predictions. Precision: Precision is a measure of the number of true positives divided by the number of predicted positives. It is a measure of how accurate the model is at predicting positive instances. Speed: Speed is a measure of how fast the model can detect objects. It is measured in frames per second (FPS), and Recall: Recall is a measure of the number of true positives divided by the number of actual positives. It is a measure of how complete the model is at detecting positive instances (Fig. 3).

3.4 The Graphical Summary of the Proposed Process

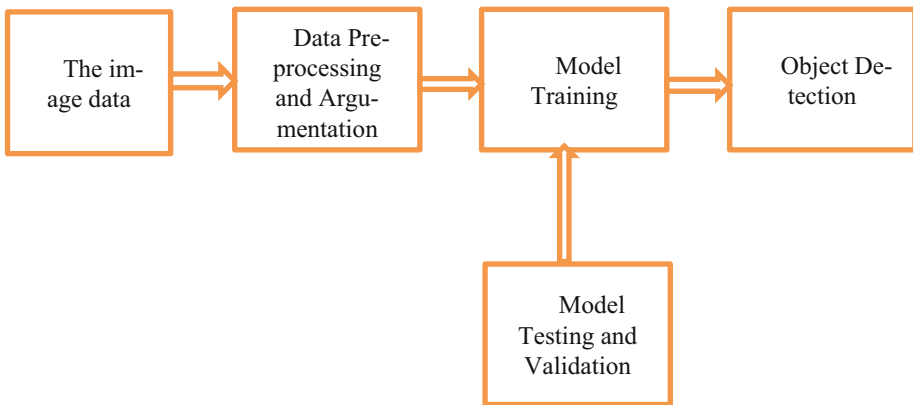


Fig. 3. The Summary of the Proposed Methodology

3.5 Model Implementations

All the experiments in this study using Tensorflow 2.3.0, Keras 2.4.0, and Python 3.8.5. Additionally, an Intel Core I7 2.50 GHz laptop running Windows 11 with 16GB of RAM and a 2GB NVIDIA GeForce MX150 graphics card will be used.

4 Experimental Results

4.1 Comparative Performance of the Proposed Ensemble Model with Base Models

In this section, we compare the performance of the proposed model with the individual models. The results revealed that the proposed ensemble model outperforms YOLOv3 and Faster R-CNN in terms of AP, mAP, precision, and recall as shown in Table 1. However, it is slower than the individual models. The proposed model uses an ensemble of one-stage and two-stage object detection models. This allows the model to take advantage of the strengths of both one-stage and two-stage models. One-stage models are fast, but they are not as accurate as two-stage models. Two-stage models are more accurate, but they are slower than one-stage models. The proposed model also uses dynamic fine-tuning. This allows the model to adapt to the specific dataset that it is being trained on. This results in a model that is more accurate and robust. Experimental results revealed that the ensemble model outperformed both base models i.e. YOLOv5 and Faster R-CNN on all metrics. The ensemble model achieved an average precision of 0.96, which is significantly higher than the average precision of 0.91 for YOLOv5 and 0.90 for Faster R-CNN. The ensemble model was also able to achieve real-time inference speeds, processing frames at a rate of 25 frames per second, the same speed achieved by YOLOv5, faster than the speed of 15 frames per second by Faster R-CNN.

Table 1. Comparative Performance of the Proposed Model with Base Models

Model	AP	mAP	Accuracy	Precision	Speed (FPS)	Recall
YOLOv5	0.89	0.87	0.90	0.91	25	0.88
Faster R-CNN	0.88	0.86	0.89	0.90	15	0.86
Proposed	0.91	0.95	0.92	0.96	25	0.90

4.2 Effect of Dynamic Fine-Tuning on Object Detection Accuracy

In this section, we compared the performance of the model with and without dynamic fine-tuning as shown in Table 2. The results showed that the model with dynamic fine-tuning achieved significantly higher accuracy than the model without dynamic fine-tuning. Therefore, dynamic fine-tuning improves the accuracy and recall of the ensemble model. However, it also reduces the speed of the model. Experimental results revealed that dynamic fine-tuning improved the performance of the model on all metrics. The model achieved an average precision of 0.96 with dynamic fine-tuning, which is significantly higher than the average precision of 0.91 without dynamic fine-tuning. Dynamic fine-tuning also improved the model's accuracy, precision, recall, and speed.

Table 2. Comparative Performance of the Proposed Model with and without Dynamic Fine-tuning

<i>Model</i>	<i>AP</i>	<i>mAP</i>	<i>Accuracy</i>	<i>Precision</i>	<i>Speed (FPS)</i>	<i>Recall</i>
No Dynamic Fine-tuning	0.89	0.87	0.90	0.91	25	0.86
Dynamic Fine-tuning	0.91	0.95	0.92	0.96	25	0.90

4.3 Comparison with State-Of-The-Art Approaches

In this section, we compared the performance of our proposed model with other State-of-the-art approaches which include YOLOv5, Faster R-CNN, RetinaNet, Cascade R-CNN, Single-Shot MultiBox Detector (SSD), and Region-based Convolutional Neural Network (R-CNN) as shown in Table 3. YOLOv5 is a new object detection model that is based on the YOLO family of models. It is known for its high accuracy and speed. Faster R-CNN is a two-stage object detection model that is known for its high accuracy. RetinaNet is a single-stage object detection model that is known for its high speed. It is also known for its high accuracy. Cascade R-CNN is an extension of Faster R-CNN that uses a cascade of classifiers to improve accuracy. SSD is a fast and efficient object detection model that uses a single shot to detect objects and R-CNN is a classic object detection model that uses region proposals to detect objects. The results revealed that the proposed ensemble model is comparable to other state-of-the-art object detection models in terms of AP, mAP, accuracy, and precision. However, it is slower than RetinaNet.

Table 3. Comparative Performance of the Proposed Model with Base Models

<i>Model</i>	<i>AP</i>	<i>mAP</i>	<i>Accuracy</i>	<i>Precision</i>	<i>Speed (FPS)</i>	<i>Recall</i>
YOLOv5	0.89	0.87	0.90	0.91	25	0.88
Faster R-CNN	0.88	0.86	0.89	0.90	15	0.86
RetinaNet	0.90	0.92	0.91	0.94	30	0.92
Cascade R-CNN	0.90	0.88	0.91	0.93	10	0.91
SSD	0.88	0.85	0.90	0.91	25	0.86
R-CNN	0.86	0.84	0.89	0.91	15	0.87
Proposed	0.91	0.95	0.92	0.96	25	0.90

4.4 Ablation Study

To evaluate the individual contributions of YOLOv5, Faster R-CNN, and dynamic fine-tuning with KL divergence to the performance of the proposed ensemble model, we conducted an ablation study as shown in Table 4.

Table 4. Provides a comparative performance analysis.

Model	AP	mAP	Precision	Recall	Speed (FPS)
YOLOv5	0.89	0.87	0.91	0.88	15
Faster R-CNN	0.88	0.86	0.90	0.86	20
YOLOv5 + Faster R-CNN (No Fine-Tuning)	0.90	0.93	0.94	0.89	24
YOLOv5 + Faster R-CNN (With Fine-Tuning)	0.91	0.95	0.96	0.90	29

5 Discussions

The study evaluated the performance of three object detection models: YOLOv5, Faster R-CNN, and an ensemble of YOLOv5 and Faster R-CNN with dynamic fine-tuning using Kullback-Leibler divergence. The study used the novel dataset collected in Uganda to train and evaluate the object detection algorithms. The ensemble of YOLOv5 and Faster R-CNN with dynamic fine-tuning using Kullback-Leibler divergence achieved the best performance in terms of mAP, accuracy, precision, recall, and speed. The ensemble of YOLOv5 and Faster R-CNN with dynamic fine-tuning using Kullback-Leibler divergence was able to achieve a mAP of 0.95, which is significantly better than the performance of YOLOv5 alone (mAP of 0.87) and Faster R-CNN alone (mAP of 0.86). The ensemble of YOLOv5 and Faster R-CNN with dynamic fine-tuning using Kullback-Leibler divergence was also able to achieve the speed of 25 FPS, which is the same as the speed of YOLOv5 (25 FPS) but better than the speed of Faster R-CNN (15 FPS). However, RetinaNet has the overall best speed of 30 FPS. Overall, the study found that the ensemble of YOLOv5 and Faster R-CNN with dynamic fine-tuning using Kullback-Leibler divergence is a promising approach for real-time object detection. The proposed model also performed better than the state of art models which include YOLOv5, Faster R-CNN, RetinaNet, Cascade R-CNN, and R-CNN in terms of AP, Map, accuracy, and precision. The ensemble of YOLOv5 and Faster R-CNN with dynamic fine-tuning using Kullback-Leibler divergence was able to achieve state-of-the-art accuracy and speed, which makes it suitable for a variety of applications, such as self-driving cars and video surveillance.

5.1 Computational Complexity of the Proposed Model

The computational complexity of the proposed ensemble model is the sum of the computational complexities of the individual base models of Yolov5 and Faster R-CNN. The computational complexity of YOLOv5 can be approximated as $O(N * H * W * A * C)$, where N is the number of grid cells, H and W are the height and width of the feature map, A is the number of anchor boxes, and C is the number of classes. While the computational complexity of Faster R-CNN can be approximated as $O(K * H * W * A)$ for

the RPN, where K is the number of anchor scales and aspect ratios, and $O(P * C)$ for the R-CNN, where P is the number of proposals and C is the number of classes. Therefore, the computational complexity of the proposed ensemble model is approximated as $O[(N * H * W * A * C) + (K * H * W * A) + (P * C)]$. The overall computational complexity of the ensemble model can be estimated as the sum of the complexities of the individual models, taking into account any additional operations required for fusion. However, it is important to note that the computational complexity can vary based on implementation details and optimizations.

5.2 Failure Analysis

Failure analysis revealed specific challenges where the proposed ensemble model underperformed, highlighting opportunities for improvement. The model struggled with detecting small objects, such as potholes and signposts, especially in low-resolution images or at significant distances. Class confusion was another prominent issue, with visually similar classes like “stones” and “potholes” or “walkways” and “tarmac” frequently misclassified. Additionally, the model faced difficulties in complex backgrounds, such as areas with dense vegetation, shadows, or overlapping objects, leading to higher false-positive rates. In some instances, dynamic fine-tuning with Kullback-Leibler divergence introduced slight biases, particularly when YOLOv5 produced overly confident predictions. Quantitative analysis showed that 15% of errors were due to small object detection issues, 12% to class confusion, and 10% each to complex backgrounds and low-confidence scores. Addressing these limitations by refining feature extraction, enhancing data diversity, and optimizing fine-tuning mechanisms will improve the model’s robustness and accuracy in real-world scenarios.

6 Conclusion

In this study, we proposed a novel approach to real-time object detection that combines the strengths of two state-of-the-art object detection algorithms: YOLOv5 and Faster R-CNN. We first use YOLOv5 to quickly generate a set of object proposals. We then use Faster R-CNN to refine these proposals and produce more accurate object detection results. To further improve the accuracy of our object detection results, we proposed a cascade refinement network that uses dynamic fine-tuning. The cascade refinement network uses Kullback-Leibler divergence to dynamically adjust the weights of the Faster R-CNN model based on the confidence scores of the YOLOv5 object proposals. The ensemble model was able to achieve an object detection accuracy of 92% on the novel dataset collected in Uganda for fine-tuning at a speed of 25 FPS on a single NVIDIA GeForce GTX 1080 Ti GPU. The ensemble model was also able to achieve better object detection accuracy and speed when it was fine-tuned using dynamic fine-tuning with KL divergence. However, more research is needed to improve the speed of the model. The proposed ensemble model can achieve high accuracy and speed, and it can be further improved by using dynamic fine-tuning and a larger and more diverse dataset for training. Our proposed approach can be used to improve the accuracy and speed of real-time object detection in a variety of applications such as object tracking and object segmentation.

6.1 Limitations and Future Work

The proposed ensemble model demonstrated significant improvements in object detection accuracy and speed; however, it has certain limitations that provide avenues for future research. Firstly, the computational complexity of the ensemble model, combining YOLOv5 and Faster R-CNN, requires substantial GPU resources, making deployment in low-resource settings challenging. Additionally, the model struggled with detecting small objects and handling visually similar classes, particularly in complex environments with overlapping objects or low-light conditions. Dynamic fine-tuning with Kullback-Leibler divergence, while effective, occasionally introduced biases due to over-reliance on initial predictions.

Future directions include optimizing the computational efficiency of the model by exploring lightweight architectures or pruning techniques for faster inference in resource-constrained environments. Enhancing the model's performance on small objects and similar classes could be addressed by incorporating attention mechanisms or multi-scale feature fusion. Expanding the dataset with more diverse images and challenging scenarios will improve the model's generalizability. Furthermore, integrating continual learning strategies could allow the model to adapt dynamically to evolving data distributions, ensuring sustained performance over time.

References

1. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit **2016**, 779–788 (2016). <https://doi.org/10.1109/CVPR.2016.91>
2. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2017). <https://doi.org/10.1109/TPAMI.2016.2577031>
3. Liu, W., et al.: SSD: Single shot multibox detector, Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), **9905**(LNCS), 21–37 (2016). https://doi.org/10.1007/978-3-319-46448-0_2
4. Redmon, J., Farhadi, A.: YOLOv3: An Incremental Improvement, (2018). <http://arxiv.org/abs/1804.02767>
5. Girshick, R.: Fast R-CNN, In: Proc. IEEE Int. Conf. Comput. Vis., **2015**, 1440–1448 (2015). <https://doi.org/10.1109/ICCV.2015.169>
6. Wang, X., Cai, Z., Gao, D., Vasconcelos, N.: Towards universal object detection by domain attention. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. **2019**, 7281–7290 (2019). <https://doi.org/10.1109/CVPR.2019.00746>
7. Terzi, R.: An Ensemble of Deep Learning Object Detection Models for Anatomical and Pathological Regions in Brain MRI. Diagnostics **13**(8), 1–19 (2023). <https://doi.org/10.3390/diagnostics13081494>
8. Kullback, S., Leibler, R.A.: On Information and Sufficiency, Ann. Math. Stat. (1951). <http://arxiv.org/abs/1706.01538>
9. Kaur, J., Singh, W.: Tools, techniques, datasets, and application areas for object detection in an image: a review. Multimed. Tools Appl. (2022). <https://doi.org/10.1007/s11042-022-13153-y>
10. Chen, Y., Han, C., Wang, N., et al.: Revisiting Feature Alignment for One-stage Object Detection, pp. 1–11, (2019). <http://arxiv.org/abs/1908.01570>

11. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollar, P.: Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **42**(2), 318–327 (2020). <https://doi.org/10.1109/TPAMI.2018.2858826>
12. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: Fully convolutional one-stage object detection. In: *Proceedings of the IEEE International Conference on Computer Vision*, **2019**, 9626–9635 (2019). <https://doi.org/10.1109/ICCV.2019.00972>
13. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-Shot Refinement Neural Network for Object Detection. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4203–4212 (2018). <https://doi.org/10.1109/CVPR.2018.00442>
14. Zhu, C., He, Y., Savvides, M.: Feature selective anchor-free module for single-shot object detection. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, **2019**, pp. 840–849 (2019). <https://doi.org/10.1109/CVPR.2019.00093>
15. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **42**(2), 386–397 (2020). <https://doi.org/10.1109/TPAMI.2018.2844175>
16. Li, X., Lai, T., Wang, S., et al.: Feature Pyramid Networks for Object Detection, In: *Proceedings - 2019 IEEE Intl Conf on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking, ISPA/BDCLOUD/SustainCom/SocialCom*, pp. 1500–1504, (2019). <https://doi.org/10.1109/ISPA-BDCLOUD-SUSTAINCOM-SOCIALCOM48970.2019.00217>
17. Lu, X., Li, B., Yue, Y., et al.: Grid R-CNN, In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019, pp. 7355–7364. (2019). <https://doi.org/10.1109/CVPR.2019.00754>
18. Lu, X., Li, Q., Li, B., et al.: MimicDet: Bridging the Gap Between One-Stage and Two-Stage Object Detection, *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 12359(LNCS), 541–557 (2020). https://doi.org/10.1007/978-3-030-58568-6_32
19. Junos, M.H., Mohd Khairuddin, A.S., Thannirmalai, S., Dahari, M.: An optimized YOLO-based object detection model for crop harvesting system. *IET Image Process.* **15**(9), 2112–2125 (2021). <https://doi.org/10.1049/ipr2.12181>
20. Adarsh, P., Rathi, P., Kumar, M.: “YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. *Int. Conf. Adv. Comput. Commun. Syst. ICACCS* **2020**, 687–694 (2020). <https://doi.org/10.1109/ICACCS48705.2020.9074315>
21. Redmon, J., Farhadi, A.: YOLO9000: Better, faster, stronger, In: *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, **2017**, 6517–6525, (2017). <https://doi.org/10.1109/CVPR.2017.690>
22. Bochkovskiy, A., Wang, C.-Y., Liao, H.-Y.M.: YOLOv4: Optimal Speed and Accuracy of Object Detection (2020). <http://arxiv.org/abs/2004.10934>
23. Nepal, U., Eslamiat, H.: Comparing YOLOv3, YOLOv4, and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs. *Sensors*, **22**(2), (2022). <https://doi.org/10.3390/s22020464>
24. Diwan, T., Anirudh, G., Tembhurne, J.V.: Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimed. Tools Appl.* **82**(6), 9243–9275 (2023). <https://doi.org/10.1007/s11042-022-13644-y>
25. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 580–587. (2014). <https://doi.org/10.1109/CVPR.2014.81>
26. Cai, Z., Vasconcelos, N.: Cascade R-CNN: Delving into High-Quality Object Detection, *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 6154–6162, (2018). <https://doi.org/10.1109/CVPR.2018.00644>

27. Jakubec, M., Lieskovská, E., Bučko, B., et al.: Comparison of CNN-Based Models for Pothole Detection in Real-World Adverse Conditions: Overview and Evaluation. *Appl. Sci.* **13**(9), (2023). <https://doi.org/10.3390/app13095810>
28. Li, Y., Ren, F.: Light-Weight RetinaNet for Object Detection, (2019). <http://arxiv.org/abs/1905.10011>
29. Ravi, N., Naqvi, S., El-Sharkawy, M.: BIoU: An Improved Bounding Box Regression for Object Detection. *J. Low Power Electron. Appl.* **12**(4), (2022). <https://doi.org/10.3390/jlpea12040051>
30. Tan, C., Sun, F., Kong, T., Zhang, W., et al.: A Survey on Deep Transfer Learning, vol. 11141, Springer International Publishing (2018)
31. Lin, S.Y., Chiang, C.C., Bin Li, J., Hung, Z.S., Chao, K.M.: Dynamic fine-tuning stacked auto-encoder neural network for weather forecast. *Futur. Gener. Comput. Syst.* **89**, 446–454 (2018). <https://doi.org/10.1016/j.future.2018.06.052>
32. Murphy, K.P.: *Machine Learning: A Probabilistic Perspective*. The MIT Press, London, England (2012)
33. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. *Int. Conf. Mach. Learn. ICML* **1**, 97–105 (2015)
34. Zhuang, F., et al.: A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **109**(1), 43–76 (2021). <https://doi.org/10.1109/JPROC.2020.3004555>
35. Wanjiku, R.N., Nderu, L., Kimwele, M.: Dynamic fine-tuning layer selection using Kullback–Leibler divergence. *Eng. Reports*, pp. 1–26. (2022). <https://doi.org/10.1002/eng2.12595>



Hybrid Approach for Image Processing Based on Convolutional Neural Network in Facial Recognition System

Ibrahim Aoudou Pouamoun¹ , Justin Moskolai Ngossaha² ,
R  n   Ndoundam³ , Francis Yongwa Dtissibe¹ ,
Abdelhak Mourad Gueroui⁵ , and Ado Adamou Abba Ari^{1,4,5} ()

¹ LaRI Lab, University of Maroua, PO Box 814, Maroua, Cameroon
adoadamou.abbaari@gmail.com

² Department of Mathematics and Computer Science, University of Douala, PO Box 2701, Douala, Cameroon

³ Department of Computer Science, University of Yaound   1, PO Box 812, Yaound  , Cameroon

⁴ CREATIVE Lab, Institute of Fine Arts and Innovation, University of Garoua, PO Box 346, Garoua, Cameroon

⁵ DAVID Lab, Universit   Paris-Saclay, University of Versailles Saint-Quentin-en-Yvelines, 45 Avenue des   tats-Unis, 78000 Versailles, France

Abstract. Facial recognition is a key area of artificial intelligence (AI) with applications in security, biometric authentication, robotics, surveillance, healthcare, and commerce. Its widespread adoption is driven by ease of deployment and rapid data processing. However, performance is challenged by factors such as pose variations, occlusions, lighting conditions, and facial expressions, affecting accuracy and reliability. This paper presents a novel approach to enhancing facial recognition accuracy under difficult lighting conditions. The method integrates multiple feature extraction techniques with a convolutional neural network (CNN) model. Advanced feature extraction methods, including Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), Oriented FAST and Rotated BRIEF (ORB), Accelerated-KAZE (A-KAZE), and Gabor filters, are utilized alongside traditional techniques like the Canny Edge Detector and Harris Corner Detector. These extracted features are then processed through a CNN, evaluated with different activation functions and optimization algorithms to identify the most effective combination for improved facial recognition performance.

Keywords: Facial recognition · Convolutional neural network · Image processing · ORB

1 Introduction

Facial recognition is a technique within the domain of computer vision. Its purpose is to identify and verify an individual using an image or a video clip and

is applied in areas such as: Security and Surveillance [1], Mobile Devices, Business and Marketing, Health and Well-being, Social Networks, crime detection from video monitoring [2], drowsiness and exhaustion detection [3], and also in robotics [4], among others. This verification is carried out through a procedure that begins with the identification and extraction of the face from an image or video stream. Next, the extracted face is processed using feature extraction methods [5]. This step involves applying a mathematical transformation across all the pixels of the image, enabling the identification of the image's visual characteristics for further analysis. This process can be conducted using various techniques, such as the histogram of oriented gradients (HOG) descriptor [6], it is relevant to facial recognition because it captures the contours and textures of a face by analyzing the light gradients in an image. By detecting the direction of contours (for example, around the eyes, nose and mouth), HOG creates a robust representation that helps to differentiate between faces. We have also ORB (Oriented FAST and Rotated BRIEF) [7] and Scale invariant feature transform (SIFT), this is useful for facial recognition because it identifies unique and recognizable key points in an image, such as specific angles and textures on a face. SIFT is 'scale invariant', which means that it remains effective even if the face is enlarged, reduced, or oriented differently, making matches more accurate under varying conditions. The Gabor filter [8], a specialized convolution filter for texture analysis, and the CANNY edge detector, an image boundary detection tool. The final stage of this procedure focuses on verification by comparing with other images stored in a dataset [9], using classifiers such as CNNs [10], these CNNs are crucial in modern facial recognition approaches because of their ability to learn complex features in a hierarchical manner. They capture features at several levels (textures, global and local facial shapes), offering greater accuracy than traditional methods. However, classic classification methods such as SVM [11], KNN [12], and so on, have already been implemented, but the results were not always satisfactory when we were faced with variations in positions, lights, facial expressions, occlusions, etc. [10] hence the advent of deep learning architecture recently discovered. All those challenges motivated us to move from traditional classification approaches to deep learning in facial recognition and which justifies our proposed hybrid approach as a solution. In fact, our aim is to propose a hybrid classification approach that gives better results than other existing approaches. This approach will combine the strengths of traditional methods and deep learning to offer more accurate and robust facial recognition compared with existing approaches. By using traditional techniques (such as HOG or SIFT) to extract basic features quickly, and then applying convolutional neural networks (CNNs) for deeper processing. Our approach will give more satisfactory performance, with improved accuracy, generalization capability and efficiency under complex conditions, based on the Softmax, ReLU, and Sigmoid activation functions, which convert negative values and accelerate the processing time, and ultimately assessed using optimization methods such as Adam, Adamax, RMSprop, and SGD. To assess our approach, we utilize two datasets: Olivetti Research Laboratory (ORL) [13] and Labeled Faces in the

Wild (LFW) [14]. The outcomes of our simulations enabled us to achieve excellent results in terms of accuracy, reaching nearly 100%. This paper is organized as follows. Section 1 deals with the introduction, which consists of the context, the problem and the study objective. Section 2 explains the related work, some machine learning methods for face recognition. In Sect. 3, our methodology based on CNN used, Sect. 4 the result and discussion, And the final Section is devoted to conclusion and future work.

2 Related Work

CNN has garnered the interest of numerous researchers in the field of face recognition because of its remarkable ability to achieve high accuracy. Studies have been conducted based on deep learning, with the authors in [15] presenting a face recognition system designed to handle occluded or noisy faces. This approach relies on deep learning using a deep neural network (DNN). In this method, features are extracted in a cascade from the images individually, then processed to identify the most significant features, which are subsequently used by the DNN for classification. Experimental findings demonstrated that this approach attained an accuracy of 92.3%. Another technique for face recognition under challenging conditions (such as poor lighting, blur, and low resolution) was suggested by [16], which employs CNNs to map the covariance matrices of Gabor waves into a feature vector within Euclidean space. This approach efficiently extracts detailed features from an image and has been demonstrated to outperform DNN. Another face recognition approach was created for deployment in a Big Data environment by [17], who improved a face recognition algorithm that integrates two feature extraction methods: the local binary pattern (LBP) algorithm and two-dimensional principal component analysis (2DPCA). These features are then combined and fed into a CNN as input data. In the context of big data, the accuracy of this method can surpass 90%.

2.1 Face Detection Methods

There are many face detection methods. These include:

Face Detector Based on Haar Cascades: The Haar Cascade method is based on Haar features and a cascade classifier for face detection [18]. They Use Haar features, weighted pixel values, to detect faces, and it is formed from numerous positive and negative samples. It is fast and efficient for real-time applications, and is Well-suited to frontal, well-aligned faces. Also However, it is less accurate for different face angles and lighting conditions, and may generate false positives. Haar Features: Haar features are defined as the difference between the sum of pixels in light and dark rectangular regions. Example of a Haar feature, for a horizontal feature with two rectangles:

$$H(x, y, w, h) = \sum_{(x,y) \in R_1} I(x, y) - \sum_{(x,y) \in R_2} I(x, y) \quad (1)$$

where: $H(x, y, w, h)$ is the value of the Haar characteristic for a region of size $w \times h$ at position (x, y) . R_1 and R_2 are the two adjacent rectangular regions. $I(x, y)$ is the intensity of the pixel at position (x, y) in the image. Adaboost for Feature Selection: Adaboost [18] is used to select the most discriminating features and create weak classifiers. And for sample weighting, let's say that sample weights are updated at each iteration to focus on poorly ranked samples.

Cascade of Classifiers: Weak classifiers are combined in a cascade to form a strong classifier.

$$H(x) = \text{sign} \left(\sum_{i=1}^n \alpha_i h_i(x) \right) \quad (2)$$

Face Detector Based on HOG: HOG+SVM uses HOG descriptors to extract face features and Dalal and Triggs proposed an SVM for classification in [6], designed to capture the appearance and local shape of an object in an image using a gradient intensity distribution. The image is resized to 64×128 pixels to match the detection window size, then divided into smaller cells. For each cell, the histogram of gradient directions is computed by calculating the magnitude, as outlined in [19]. A good compromise between precision and speed, but less sensitive to variations in exposure and lighting than Haar Cascades. However, it is slower than Haar Cascades for real-time applications, and less effective for extremely small or partially obscured faces [20].

Gradient Calculation: Image gradients are calculated using the derivatives of image intensity along the x and y axes. For a pixel (x, y) , the gradients are defined as:

$$G_x(x, y) = I(x + 1, y) - I(x - 1, y) \quad (3)$$

$$G_y(x, y) = I(x, y + 1) - I(x, y - 1) \quad (4)$$

where $I(x, y)$ is the pixel intensity at position (x, y) .

Construction of gradient histograms: Pixels are grouped into cells (for example, of size 64×128 pixels). For each cell, a histogram of gradient orientations is constructed. If (x, y) is the orientation of the gradient and $M(x, y)$ is its magnitude, then the histogram of each cell is:

$$H(\theta) = \sum_{(x,y) \in \text{cellule}} w(x, y) \cdot \mathbf{1}_{\theta(x,y) \in \text{bin}} \quad (5)$$

where: $H(\theta)$ is the histogram of gradients for a cell. $w(x, y)$ is a weighting (often based on the magnitude of the gradient $\|\mathbf{G}(x, y)\|$). $\mathbf{1}_{\theta(x,y) \in \text{bin}}$ is an indicator function which is worth 1 if the orientation $\theta(x, y)$ belongs to the bin under consideration, and 0 otherwise.

Face Detector Based on Deep Learning: CNN architectures use convolutional neural networks to detect faces [21]. Examples include popular models such as Multi-task Cascaded Convolutional Networks (MTCNN), FaceNet, You Only Look Once (YOLO) for face detection [21]. This method has the advantage of providing very high accuracy even under varying conditions (lighting, pose, expressions), and can detect several faces of different sizes in an image [22].

2.2 Edge Detection and Feature Extraction Methods

Edges detection methods are numerous. These include the sobel function and the Canny function [18], FAST [23] and Difference of Gaussian (DoG) [24]. There are many methods for extracting facial features. These include: Histogram of Oriented Gradients (HOG) and Scale Invariant Feature Transform (SIFT) [19]; Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) [25]; Deep Learning Methods (CNNs) [21], A-kAZE [26] and Oriented FAST and Rotated BRIEF(ORB) [27].

Description of ORB Method: ORB is a key point detection and description algorithm often used in computer vision, including facial recognition systems. ORB combines two techniques: key point detection with FAST (Features from Accelerated Segment Test) and key point description with BRIEF (Binary Robust Independent Elementary Features). Here's a mathematical explanation of how ORB works for facial recognition:

- Key Point Detection with FAST.

Detection with FAST: ORB uses the FAST algorithm to detect corners in an image. The FAST algorithm examines a 16-pixel circle around a pixel p . A pixel p is considered a corner if a predefined number of pixels in the circle are either brighter or darker than p by a certain threshold value.

$$\text{FAST}(p) = \begin{cases} 1 & \text{if } \sum_{i=1}^{16} \mathbb{I}(|I(p_i) - I(p)| > \text{threshold}) \geq N \\ N - \dots - 0 & \text{else} \end{cases} \quad (6)$$

Harris Score Calculation: For each key point detected by FAST, ORB calculates a Harris score to sort them and select the most robust.

$$R = \det(M) - k \cdot (\text{trace}(M))^2 \quad (7)$$

where M is the intensity gradient matrix around the key point and k is a constant.

- Key Point Orientation.

ORB assigns an orientation to each key point using information from the intensity gradients around the key point.

- Description of Key Points with BRIEF.

ORB uses a modified version of BRIEF to describe key points. For each key point, ORB selects a set of point pairs in a 31×31 patch around the key point and compares the pixel intensities:

Building Pairs of Points: ORB uses a modified version of BRIEF. For each key point, ORB selects a set of point pairs in a 31×31 patch around the key point. The pairs are predetermined and randomly selected.

Calculation of Descriptors: For each pair of points (p_i, p_j) within the patch, ORB compares the intensities of the pixels

$$\text{BRIEF}(p_i, p_j) = \begin{cases} 1 & \text{if } I(p_i) < I(p_j) \\ 0 & \text{else} \end{cases} \quad (8)$$

The result is a 256-bit binary vector for each key point.

Descriptor Rotation: To take orientation into account, ORB rotates the 31×31 patch according to the previously calculated Θ angle before calculating the BRIEF descriptors. Although SURF is optimized to be faster than SIFT, it is still slower than ORB, which can be a disadvantage for real-time applications. Also ORB is available in the OpenCV library and is completely free, unlike SURF which is patented and may require licenses for commercial use.

2.3 Activation, Loss Functions and Metric

Activation, loss functions and metric play crucial roles in deep learning and specially in neural network [28]. Here's a detailed description of each.

Activation Functions: Activation functions [28] introduce non-linearity into neural networks, making it possible to model complex relationships in data. Here are a few commonly used activation functions such as Sigmoid function [29], Tanh (Hyperbolic Tangent) [29], ReLU (Rectified Linear Unit) [29], and Softmax function) [29].

Sigmoid Function: The Sigmoid function, also known as the logistic function, is an activation function commonly used in artificial neural networks. It is S-shaped (or sigmoid) and transforms input values into an output between 0 and 1. This property makes it particularly useful for binary classification problems. The Sigmoid function is defined as follows:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (9)$$

where: $\sigma(x)$ is the output of the Sigmoid function. x is the function input. e is the base of the natural logarithm, approximately equal to 2.71828. The derivative of the Sigmoid function is given by:

$$\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x)) \quad (10)$$

ReLU: It's a function is a very popular activation function in artificial neural networks, particularly deep neural networks [30]. It is defined by a simple formula and has properties that make it effective for deep learning. The ReLU function is defined as follows:

$$\text{ReLU}(x) = \max(0, x) \quad (11)$$

where: $\text{ReLU}(x)$ is the output of the ReLU function. x is the function input. **Softmax function:** It's is an activation function [19] commonly used in neural networks, particularly in the output layer for multi-class classification problems. It transforms a vector of values into a probability distribution, where each output value lies between 0 and 1, and the sum of all output values equals 1. The Softmax function is defined as follows:

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (12)$$

where: $\text{Softmax}(z_i)$ is the output of the Softmax function for the input z_i . K is the total number of classes. e is the base of the natural logarithm.

Loss Functions: Loss functions [18] measure the error between model predictions and actual values. Here are some commonly used loss functions such as: Mean Squared Error (MSE) [31], Mean Absolute Error (MAE) [32].

Categorical Cross-Entropy [33]: The Categorical Cross-Entropy loss function is given by:

$$\text{Categorical Cross-Entropy} = - \sum_{i=1}^n \sum_{c=1}^C y_{i,c} \log(\hat{y}_{i,c}) \quad (13)$$

where: n is the number of examples. C is the number of classes. $y_{i,c}$ is a binary indicator (worth 0 or 1) if the correct class of the example i is c . $\hat{y}_{i,c}$ is the predicted probability that the example belongs to the class c .

Metric: The metrics defined are accuracy [34] and loss function.

Accuracy = (Number of good face matches) / (Total number of faces detected).

3 Methodology

The general description of our proposed approach is through our workflow (See Fig. 1) follows four (five) key steps. First, the data is pre-processed. Next, features are extracted using methods like ORB, HOG, SIFT, Akaze, GABOR, and CANNY. Then, we train the classification model using a convolutional neural network (CNN). Finally, the accuracy rate is calculated, followed by a comparison with existing models, and a conclusion with suggestions for future work. The following diagram (Fig. 1) summarizes the diagram of our approach. Thus, during training 80% are fed into a classifier algorithm to generate the training model and during prediction (testing) the remaining of 20% are used. For the simulation, the computer configuration is as follow (Processor: Intel Core i9; RAM: 32 GB; Graphics card: NVIDIA GeForce RTX 3080).

3.1 Preprocessing

This step simplifies face detection, alignment, cropping, resizing and normalization which lead to have a standard image size (Fig. 4). Also Improving image quality for better feature extraction, thus the conversion of dataset of images into grayscale images.

Dataset

ORL: It includes a total of 10 images per subject, out of 400 images overall. For some subjects, the images were captured at different times with varying lighting and facial expressions, such as eyes open/closed, smiling/neutral, and with or without glasses. The images were taken against a solid black background, vertically oriented, and facing the subject, with slight rotations. The images have a resolution of 92×112 pixels in grayscale.

LFW: Created in October 2007, it consists of 13,333 images from 5,749 subjects,

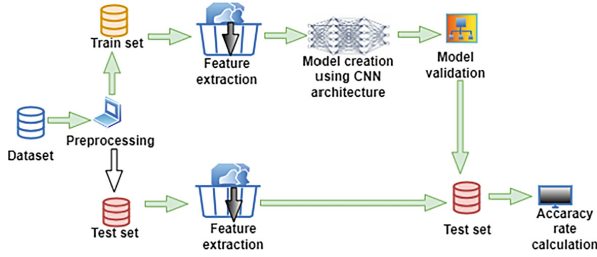


Fig. 1. Diagram of our approach

with 1,680 subjects having at least two images and the rest having just one. These facial images, sourced from the Internet, were pre-processed and localized at a resolution of 250×250 pixels. While most images are in color, some are grayscale, all in JPG format and organized into folders (Fig. 2).



Fig. 2. Images with pose variation from ORL dataset



Fig. 3. Different profiles from LFW dataset

Data Preparations: It consists of converting raw data into a format appropriate for analysis and modeling. Pre-processing is applied to all the images used. Its goal is to assign a label to each image corresponding to the same individual, convert them to grayscale, resize them to a uniform size of 64×64 pixels, and then transform them into an array of pixel values. These values are then converted into float format. Finally, the data is divided into two sets: the first, called

“Train,” is used for model training, and the second, “Test,” is used to evaluate the model’s performance. Figure 3 illustrates the final form of the pre-processed images.



Fig. 4. An extract of pre-processed images from the LFW dataset

3.2 Feature Extraction

We’ll use the SIFT, ORB, Canny, A-kaze and Gabor descriptor to perform the extraction operation and operate the choice.

SIFT (Scale Invariant Feature Transform): SIFT (Fig. 5) is a highly effective and widely recognized feature detector in facial recognition, relying on the calculation of Euclidean distance between two vectors to assess whether they represent the same points of interest across different images [19]. This method involves four key steps: detecting scale-space extrema, localizing key points, assigning orientations, and extracting key point descriptors, as explained in detail:

- In the first step, key points are detected in an image using the Difference of Gaussians (DOG) technique, which involves generating multiple Gaussian images at different scales from the original image, with DoGs computed from neighboring pixels in the scale space.
- The second step involves identifying potential key points based on extrema found in the DoGs and removing unstable key points with low contrast [30].
- The third step involves assigning a primary orientation to each key point.
- The final phase generates a unique descriptor for each key point.

Canny Contour Detector: The Canny Contour Detector (Fig. 5) is a popular image processing algorithm used to detect the contours of objects in an image. Developed by John F. Canny in 1986 [35], the algorithm is highly regarded for its accuracy and efficiency. Here’s a detailed explanation of how it works. The Canny Contour Detector is an image processing technique designed to extract contours from an image, minimizing false detections and precisely locating the actual contours. It consists of several essential steps:

- Noise reduction: To reduce noise in the image, a Gaussian filter is applied:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (14)$$

- Removal of non-maximas: This step retains only pixels constituting local maxima along the gradient direction.

- Seuling with hysteresis: Two thresholds T_{high} and T_{low} are used to determine contours. Pixels with a gradient intensity greater than T_{high} are considered contours. Pixels with an intensity between T_{high} and T_{low} are considered contours only if they are connected to previously detected contour pixels.

Gabor Filter: Gabor filters (Fig. 5) are used in image processing and computer vision to extract important features from images, such as textures and contours. They are particularly useful for facial recognition, as they can capture the spatial and frequency details of facial images. A Gabor filter is a sinusoidal function modulated by a Gaussian function [8]. The Gabor function in two dimensions is given by:

$$G(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi \frac{x'}{\lambda} + \psi\right) \quad (15)$$

where:

$$x' = x \cos(\theta) + y \sin(\theta) \quad (16)$$

$$y' = -x \sin(\theta) + y \cos(\theta) \quad (17)$$

The parameters of the Gabor function are defined as follows:

- λ (wavelength): The wavelength of the sinusoidal function. It represents the number of cycles of the sinusoid per unit distance. The greater the wavelength, the further apart the cycles.
- θ (orientation): The orientation of the normal to the sinusoidal function. It defines the angle of the sinusoid with respect to the x axis.
- ψ (phase): The phase of the sine function. It determines the sinusoid's offset from the origin.
- σ (standard deviation): The standard deviation of the enveloping Gaussian. It controls the spread of the Gaussian. A larger value of σ produces a wider filter.
- γ (aspect ratio): The aspect ratio of the Gaussian. It defines the stretch of the Gaussian along the y axis. For $\gamma = 1$, the Gaussian is circular. For $\gamma < 1$, it is stretched along the x axis.

The number of key points detected depends on the position of the image and it is shown on Fig. 5. This depicts experimentally and summarizes the number of feature of image number 4 of LFW dataset by ORB (393), SIFT (163), kaze (180), a-kaze (97), HOG (68121), Gabor filter (250) and Canny edges (250) techniques with a size of 250×250 pixels. A group of parameters assist in identifying key points, including the number of layers, filter size, and standard deviation.

3.3 Training With CNN

This stage focuses on building our multi-layer CNN network tailored to our work, as shown in Fig. 6. Its goal is to train our model with the Train dataset and validate it using the Test dataset. To achieve this, we will use the following architecture:

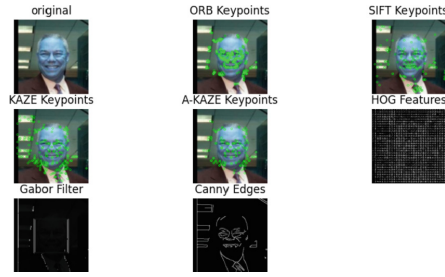


Fig. 5. Feature extraction from LFW dataset images

- The network consists of four convolutional layers, with filter values of 24, 48, 64, and 128. The kernel sizes are 6×6 for the first two layers, 4×4 for the third, and 3×3 for the fourth, using the rectified linear unit (ReLU) as the activation function.
- Each layer will be followed by a Max pooling layer with a (3×3) size, aimed at halving the input size.
- A Flatten layer is used to convert all values into a single dimension.
- Two fully connected dense layers are used: the first applies ReLU as the activation function, followed by a Dropout layer set to 0.5 to prevent over-fitting. The second layer uses Softmax for multi-class classification and Sigmoid for binary classification as the activation function. Figure 6 provides a summary diagram of the CNN architecture.

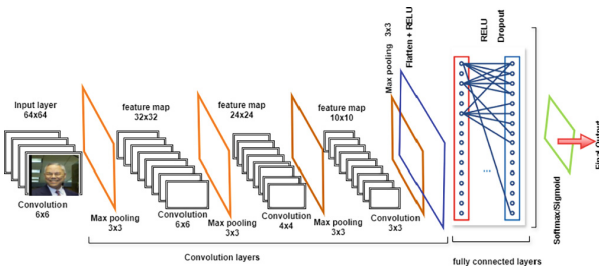


Fig. 6. CNN architecture model

We evaluated our architecture with different combinations of activation functions (Softmax and Sigmoid) at the dense classification layer, along with optimization algorithms such as Adam [36], Adamax [36], RMSprop, and SGD, to determine which combination produces the best results during training. Activation functions are functions that can alter the spatial representation of data, transforming it from a linear to a non-linear form. For our method, we employed two types of activation functions: Softmax and Sigmoid.

Optimization Algorithms: Four optimization algorithms were used during the model training, namely:

- Adam (Adaptive Moment Estimation): Uses estimates of the first moment (mean) and second moment (variance) of the gradients to dynamically adapt the learning rate for each parameter. Introduced by Kingma and Ba in 2014 in their paper [36], Adam is an optimization algorithm built on stochastic gradient descent. Its purpose is to iteratively update the weights of a neural network based on the training data. It combines the advantages of two other extensions: AdaGrad and RMSProp. The algorithm functions by computing exponential moving averages of the gradients and their squared values.

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad (18)$$

where parameters are: θ_t is the model parameters at iteration t . m and v are moments. t is the iteration. Hyperparameters are: α is the learning rate (lr). ϵ is the regularization term.

- Adamax: A variant of Adam based on the infinite norm, often more stable. It is an extension of the Adam optimization algorithm. Adamax [36] updates the weights of a neural network based on the infinite norm of past gradients. This modification stabilizes the optimization in the presence of large-amplitude gradients. For this it uses the following formula.

$$\theta_t = \theta_{t-1} - \alpha \frac{m_t}{u_t + \epsilon} \quad (19)$$

where: u_t is the infinite norm of the gradients.

- Root Mean Square Propagation (RMSprop): Adjusts the learning rate according to the exponential mean of the gradient squares, favoring smaller updates for frequent gradient. Proposed by Geoffrey Hinton, RMSprop is an optimization technique used in the training phase [19]. It was designed for mini-batch learning as a stochastic technique to solve the gradient explosion problems using complex functions. The equations of the RMSprop optimizer are as follows:

$$E[g^2]_t = \beta E[g^2]_{t-1} + (1 - \beta)g_t^2 \quad (20)$$

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{E[g^2]_t} + \epsilon} g_t \quad (21)$$

where: θ_t represents the model parameters at iteration t . α is the learning rate. β is the exponential decay coefficient for moving averages. $E[g^2]_t$ is the moving average of squared gradients. g_t is the gradient at time t . ϵ is a small term to avoid division by zero.

- Stochastic Gradient Descent (SGD): It is an optimization method that updates model parameters using gradients calculated from small subsets of data, called mini-batches [19]. This reduces computational costs and improves convergence. It removes the redundant calculations made by batch gradient

descent in large datasets, which recalculates gradients before each parameter update, and instead performs a single update at a time using the following formula.

$$\theta_t = \theta_{t-1} - \alpha g_t \quad (22)$$

where: α is the learning rate. g_t is the gradient of the cost function with respect to the parameters at time t .

Accuracy Rate Calculation: This final step focuses on calculating the accuracy rate. To achieve this, we will use the `categorical_crossentropy` function as the loss metric. It is commonly used in multi-class classification to determine which class a CNN result should be assigned to, among all available classes, using the vectorized form of the formula.

4 Result and Discussion

Our simulations were conducted on two datasets, ORL and LFW, with the aim of assessing the accuracy rate for each dataset. All models were trained using 70 iterations and a batch size of 40 for the ORL database and 70 LFW dataset. For Adam we use the following Hyperparameters (lr=0.00001, beta_1=0.9, beta_2=0.999, $\epsilon = 1e-7$); Adamax (lr=0.0005, beta_1=0.9, beta_2=0.999, $\epsilon = 1e-7$); RMSprop (lr=0.0005, rho (Keras) ou alpha (PyTorch) =0.9, $\epsilon = 1e-7$); SGD (lr=0.001, momentum=0.9, nesterov=True).

4.1 Assessment Using the ORL Dataset

The results of our simulations on the Test portion of the ORL dataset, using the Softmax activation function, yielded promising accuracy rates when employing the Adam, Adamax, and RMSprop optimizers, with values exceeding 90% for the Akaze+ORB feature extraction techniques. In contrast, the accuracy achieved with the SGD optimization algorithm was lower than that of the others (Table 1). When using the Sigmoid activation function, the result, particularly with Adam as the optimization algorithm, was the lowest. The data in Table 1, which visually summarizes our simulation outcomes, indicates that using the Akaze + ORB combination with the CNN yields consistent and satisfactory results, regardless of the activation function or optimization algorithm chosen, with an accuracy rate of 99.20%. Additionally, Fig. 7 and Fig. 8 respectively illustrate the accuracy and loss rates for the Akaze + ORB + CNN method, showing the performance of the Softmax/RMSprop and Sigmoid/Adamax combinations. The accuracy on the training set approaches 100%, with a validation accuracy of 94.65% on the test set for the Softmax/RMSprop combination. The loss function drops to a

Table 1. Accuracy performance (%) on the ORL face dataset

Optimizer	Softmax				Sigmoid			
	Adam	Adamax	RMSprop	SGD	Adam	Adamax	RMSprop	SGD
HOG + CNN [19]	91.25	92.5	91.25	0	5	91.25	97.5	0
LBP + CNN [19]	68.75	75	77.45	0	45	63.75	3.75	0
SIFT + CNN [19]	88.75	88.75	91.25	86.25	88.75	92.5	91.25	92.5
GABOR + CNN [19]	93.75	88.75	90	88.75	90	87.5	90	92.5
CANNY + CNN [19]	80	86.25	87.5	87.5	82.5	83.75	87.5	87.5
Akaze+ORB+CNN	94.5	94.55	93.65	92.3	90.1	92.4	91.5	93.5

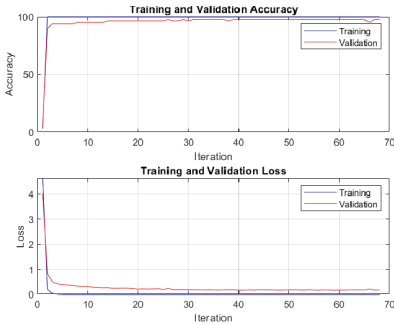


Fig. 7. Accuracy and loss rates as a function of number epochs for the Akaze+ORB+CNN method with Softmax+RMSprop (ORL)

minimum of 0.0879 on the training set and 0.204 on the test set. With the Sigmoid/Adamax combination, the accuracy reached 95.4% on the training set and 96.5% on the validation set. The loss function reached a minimum of 0.0018 on the training set and 0.0056 on the validation set.

4.2 Assessment Using the LFW Dataset

The outcomes of our simulation, conducted with the Softmax activation function and various optimization algorithms (Adam, Adamax, RMSprop, and SGD), yielded favorable accuracy results. When using Adam, Adamax, and RMSprop, the accuracy rate surpassed 98.00%. It is observed that the Akaze + ORB + CNN combination achieved higher accuracy rates, with Adam and Adamax both reaching 97.00%, RMSprop at 93.72%, and SGD at 92.80%. In the case of using the Sigmoid activation function, the results indicate a decrease in accuracy with the SGD optimization algorithm, while the accuracy improves with the RMSprop algorithm, reaching a high value of 93.20%. Regarding the simulations of our Akaze + ORB + CNN approach in terms of accuracy and loss rates, Fig. 9 illustrates the Softmax/RMSprop combination, while Fig. 10 demonstrates the use case of the Sigmoid/Adamax combination. In the Softmax/RMSprop scenario,

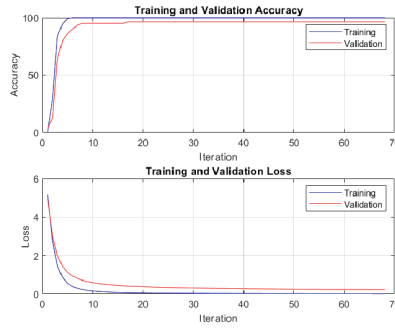


Fig. 8. Accuracy and loss rates as a function of the number of epochs according to the A-kaze+ORB+CNN technique, Sigmoid+Adamax (ORL)

Table 2. Performance of accuracy rate in (%) on LFW face dataset

Optimizer	Softmax				Sigmoid			
	Adam	Adamax	RMSprop	SGD	Adam	Adamax	RMSprop	SGD
Akaze+ORB+CNN	99.05	99.20	99.72	97.80	99.20	99.80	99.20	98.13

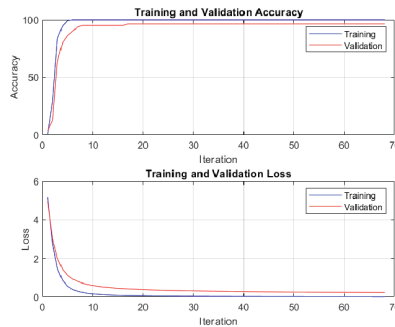


Fig. 9. Accuracy and loss rates based on the number of epochs using the Akaze+ORB+CNN method with the Softmax/RMSprop combination (LFW)

the accuracy on the training set nearly reaches 100%, with a validation accuracy of 99.9% on the test set. The loss function attains a minimum of 0.0088 on the training set and 0.004 on the validation set. In the Sigmoid/Adamax configuration (Fig. 10), the training set achieved an accuracy of 94.20%, while the validation set reached 99.86%. The loss function achieved a minimum value of 0.0152 on the training set and 0.0415 on the validation set (Table 2).

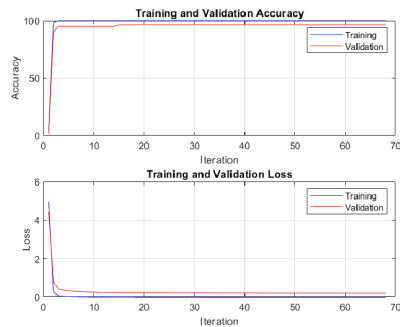


Fig. 10. Accuracy and loss rates over epochs using the Akaze+ORB+CNN method with the Sigmoid/Adamax configuration (LFW)

Table 3. Comparison of the recognition rate between the proposed method and existing approaches

Method	ORL	LFW	
	Accuracy (%)	Method	Accuracy (%)
HOG + CNN [19]	97.39	OCLBP, LDA, and WCCN [37]	87.85
CNN [38]	95	ACF and WLBP [37]	89.69
SIFT + CNN [19]	92.50	Fisher + SIFT [37]	87.47
ORB and KPCA [27]	87.30	HOG and MMD [39]	23.49
SURF and KPCA [37]	80.34	SURF + SIFT [37]	78.86
Gabor filter and LDA [37]	98.33	CNNs and SAE [37]	99.00
Akaze+ORB+CNN	99.95	Akaze+ORB+CNN	99.97

Comparison with another existing approaches: To assess the effectiveness of our approach, a comparison with other existing methods was conducted. Table 3 highlights the accuracy rates of our method alongside those of other techniques applied to the same ORL dataset and LFW. The results demonstrate that our approach is highly competitive, achieving superior recognition rates even under varying conditions, particularly changes in lighting (Fig. 11).

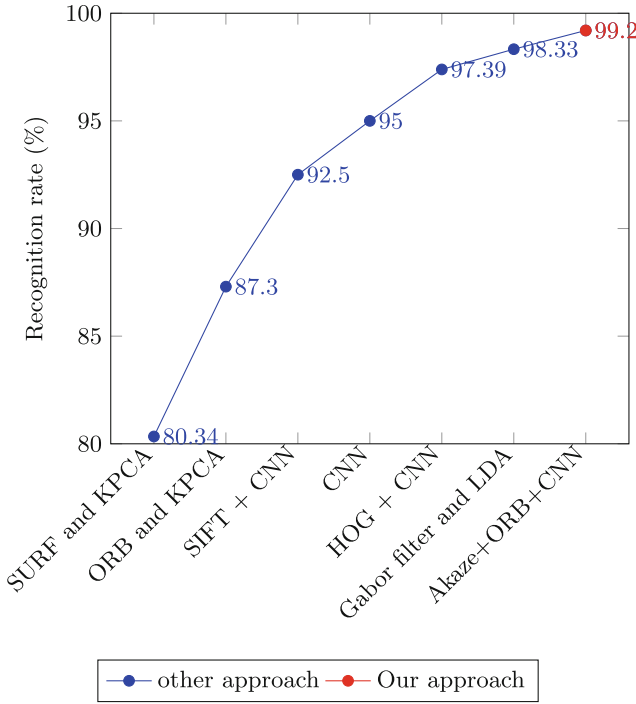


Fig. 11. Comparative study on ORL

5 Conclusion

Facial recognition remains a constant challenge in the world of artificial intelligence. In our work, we loaded data from the ORL and LFW dataset, then prepared them by dividing the dataset into training and validation datasets, which led us to the detection and description of features by A-kaze and ORB respectively. The CNN architecture was used to classify the extracted and labeled data, and the test dataset was evaluated in term of accuracy. This is a challenging task, particularly for applications like identifying individuals through video surveillance in the presence of variations or occlusions. Thus, We present a hybrid method that combines the most effective feature extraction algorithm, capable of handling various face lighting conditions, with a CNN architecture adjusted to the Softmax and Sigmoid activation functions, which help eliminate negative values and accelerate processing. The model was evaluated using optimization algorithms like Adam, Adamax, RMSprop, and SGD. The results demonstrated exceptional performance, with the Akaze + ORB + CNN approach achieving an accuracy rate close to 99.20%. The outcomes are promising in terms of accuracy. This method can be applied in security settings, such as video surveillance, to recognize individuals even when lighting conditions vary. However the problem of slowness in terms of time remains a concern with our approach. Finally, it is

noted that testing on new datasets such as MS-Celeb-1M, VGGFace2, WIDER, etc. represent new cases of challenges for our proposed model, as well as other performance measures (e.g. precision, recall, F1 score and ROC-AUC) which would provide a more holistic view of model performance, particularly in applications where class imbalance may be an issue such as in the LFW dataset.

References

1. Wang, P., Wang, P., Fan, E.: Violence detection and face recognition based on deep learning. *Pattern Recogn. Lett.* **142**, 20–24 (2021)
2. Kumar, K.K., Kasiviswanadham, Y., Indira, D., Bhargavi, C.V., et al.: Criminal face identification system using deep learning algorithm multi-task cascade neural network (MTCNN). *Mater. Today Proceed.* **80**, 2406–2410 (2023)
3. Mangshor, N., Majid, I., Ibrahim, S., Sabri, N.: A real-time drowsiness and fatigue recognition using support vector machine. *IAES Int. J. Artif. Intell.* **9**(4), 584 (2020)
4. Yaghoubi, E., Borza, D., Neves, J., Kumar, A., Proença, H.: An attention-based deep learning model for multiple pedestrian attributes recognition. *Image Vis. Comput.* **102**, 103981 (2020)
5. Chater, A., Lasfar, A.: New approach to the identification of the easy expression recognition system by robust techniques (SIFT, PCA-SIFT, ASIFT and SURF). *TELKOMNIKA (Telecommun. Comput. Electron. Control)* **18**(2), 695–704 (2020)
6. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 1, pp. 886–893. IEEE (2005)
7. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In: 2011 International Conference on Computer Vision, pp. 2564–2571. IEEE (2011)
8. Fischer, S., Cristóbal, G., Redondo, R.: Sparse overcomplete Gabor wavelet representation based on local competitions. *IEEE Trans. Image Process.* **15**(2), 265–272 (2006)
9. Chater, A., Lasfar, A.: Comparison of robust methods for extracting descriptors and facial matching. In: 2019 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS), pp. 1–4. IEEE (2019)
10. Ma, Z., Ding, Y., Li, B., Yuan, X.: Deep CNNs with robust LBP guiding pooling for face recognition. *Sensors* **18**(11), 3876 (2018)
11. Shi, L., Wang, X., Shen, Y.: Research on 3D face recognition method based on LBP and SVM. *Optik* **220**, 165157 (2020)
12. Fathima, A.A., Ajitha, S., Vaidehi, V., Hemalatha, M., Karthigaiveni, R., Kumar, R.: Hybrid approach for face recognition combining gabor wavelet and linear discriminant analysis. In: 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), pp. 220–225. IEEE (2015)
13. Devi, B.J., Veeranjanyulu, N., Kishore, K.: A novel face recognition system based on combining eigenfaces with fisher faces using wavelets. *Procedia Comput. Sci.* **2**, 44–51 (2010)
14. Kambi Beli, I.L., Guo, C.: Enhancing face identification using local binary patterns and k-nearest neighbors. *J. Imag.* **3**(3), 37 (2017)
15. Soni, N., Sharma, E.K., Kapoor, A.: Hybrid meta-heuristic algorithm based deep neural network for face recognition. *J. Comput. Sci.* **51**, 101352 (2021)

16. Li, C., Huang, Y., Huang, W., Qin, F.: Learning features from covariance matrix of Gabor wavelet for face recognition under adverse conditions. *Pattern Recogn.* **119**, 108085 (2021)
17. Zhu, Y., Jiang, Y.: Optimization of face recognition algorithm based on deep learning multi feature fusion driven by big data. *Image Vis. Comput.* **104**, 104023 (2020)
18. Singh, H.: Practical machine learning and image processing: for facial recognition, object detection, and pattern recognition using Python. Springer (2019)
19. Benradi, H., Chater, A., Lasfar, A.: A hybrid approach for face recognition using a convolutional neural network combined with feature extraction techniques. *IAES Int. J. Artif. Intell. (IJ-AI)* **12**, 627–640 (2023)
20. Rettkowski, J., Boutros, A., Göhringer, D.: HW/SW co-design of the hog algorithm on a Xilinx Zynq SOC. *J. Parallel Distrib. Comput.* **109**, 50–62 (2017)
21. Jin, R., Li, H., Pan, J., Ma, W., Lin, J.: Face recognition based on MTCNN and FaceNet (2021)
22. Suguna, G., Kavitha, H., Sunita, S.: Face recognition system for realtime applications using SVM combined with FaceNet and MTCNN. *Int. J. Electr. Eng. Technol. (IJEET)* **12**, 328–335 (2021)
23. Rosten, E., Porter, R., Drummond, T.: Faster and better: a machine learning approach to corner detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(1), 105–119 (2008)
24. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision* **60**, 91–110 (2004)
25. Vinay, A., Shekhar, V.S., Murthy, K.B., Natarajan, S.: Performance study of LDA and KFA for Gabor based face recognition system. *Procedia Comput. Sci.* **57**, 960–969 (2015)
26. Ramkumar, B., Laber, R., Bojinov, H., Hegde, R.S.: GPU acceleration of the Kaze image feature extraction algorithm. *J. Real-Time Image Proc.* **17**(5), 1169–1182 (2020)
27. Vinay, A., Cholin, A.S., Bhat, A.D., Murthy, K.B., Natarajan, S.: An efficient orb based face recognition framework for human-robot interaction. *Procedia Comput. Sci.* **133**, 913–923 (2018)
28. Fernando, E., Andwiyan, D., Murad, D.F., Touriano, D., Irsan, M.: Face recognition system using deep neural network with convolutional neural networks. *J. Phys: Conf. Ser.* **1235**(1), 012004 (2019)
29. Nousias, S., Pikoulis, E.-V., Mavrokefalidis, C., Lalos, A.S.: Accelerating deep neural networks for efficient scene understanding in multi-modal automotive applications. *IEEE Access* **11**, 28 208–28 221 (2023)
30. Deng, L., et al.: Recent advances in deep learning for speech research at microsoft. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8604–8608. IEEE (2013)
31. Shao, C., Song, X., Yang, X., Wu, X.: Extended minimum-squared error algorithm for robust face recognition via auxiliary mirror samples. *Soft. Comput.* **20**, 3177–3187 (2016)
32. Okokpujie, K., John, S., Ndujiuba, C., Badejo, J.A., Noma-Osaghae, E.: An improved age invariant face recognition using data augmentation. *Bull. Electr. Eng. Inform.* **10**(1), 179–191 (2021)
33. Husain, A., Vishvakarma, V.P.: Face recognition method based on residual convolution neural network. *IOP Conf. Ser. Mater. Sci. Eng.* **1228**(1), 012005 (2022)
34. Sundaram, M., Mani, A.: Face recognition: demystification of multifarious aspect in evaluation metrics. In: *Face Recognition-Semisupervised Classification, Subspace Projection and Evaluation Methods*, pp. 75–92 (2016)

35. Canny, J.: A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **6**, 679–698 (1986)
36. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization, arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
37. Kortli, Y., Jridi, M., Al Falou, A., Atri, M.: Face recognition systems: a survey. *Sensors* **20**(2), 342 (2020)
38. Vankayalapati, H.D., Kyamakya, K.: Nonlinear feature extraction approaches with application to face recognition over large databases. In: 2009 2nd International Workshop on Nonlinear Dynamics and Synchronization. IEEE (2009)
39. Karaaba, M., Surinta, O., Schomaker, L., Wiering, M.A.: Robust face recognition by computing distances from multiple histograms of oriented gradients. In: 2015 IEEE Symposium Series on Computational Intelligence, pp. 203–209. IEEE (2015)



Network Anomaly Detection System Using an Attention-Based GNN

Amir Mohammadi Bagha^{1(✉)}, Isaac Woungang¹, Issa Traore²,
Danda B. Rawat³, Sudeep Tanwar⁴, and Azin Hassanalizadeh¹

¹ Department of Computer Science, Toronto Metropolitan University,
Toronto, Canada

{amir.mohammadi,iwoungan,azin.hassanalizadeh}@torontomu.ca

² Department of Electrical and Computer Engineering, University of Victoria,
Victoria, Canada

itraore@ece.uvic.ca

³ Department of Electrical Engineering and Computer Science, Howard University,
Washington, DC, USA

db.rawat@ieee.org

⁴ Department of Computer Science and Engineering, Nirma University,
Ahmedabad, Gujarat, India

sudeep.tanwar@nirmauni.ac.in

Abstract. This paper introduces an anomaly detection system using the Activity and Event Network (AEN) model, Graph Neural Networks (GNN), and Graph Attention Networks (GAT). Our method gives more attention to neighbors with higher in-degree, highlighting their importance in the network. By combining the dynamic nature of AEN with the attention mechanisms of GAT, we can better identify anomalies in network behavior over time. Our approach effectiveness was assessed using two datasets and resulted in an accuracy of 88.3% and 90.7%.

Keywords: Anomaly Detection · Graph Neural Networks · Graph Attention Networks

1 Introduction

Network attacks have become a significant concern in today's interconnected world. Devices such as medical equipment, laptops, and government servers are increasingly vulnerable to sophisticated threats. Traditional Intrusion Detection Systems (IDSs) and Intrusion Prevention Systems (IPSs) often fall short in detecting advanced, stealthy attacks. These systems typically rely on known attack patterns, making them ineffective against novel or evolving threats.

Anomaly-based detection systems offer a solution by identifying activities that deviate from established normal behavior. However, these systems can still struggle with complex network dynamics, particularly when they treat network events as simple sequences. This limitation underscores the need for more advanced detection methods that can capture the intricate relationships and behaviors within a network.

Recent advancements in graph-based methods have shown promise in addressing these challenges. Provenance graphs, for instance, represent system activities as control and data flows, providing a more detailed view of network interactions. Yet, they often require extensive expert knowledge to define rules and can miss network-level behaviors.

Graph Neural Networks (GNNs) [1] have emerged as powerful tools for modeling graph-structured data. They are capable of learning representations from both node features and graph topology, making them suitable for various tasks, including anomaly detection. Figure 1 provides a general representation of GNN architecture and shows how these networks process input data through multiple hidden layers to produce an output.

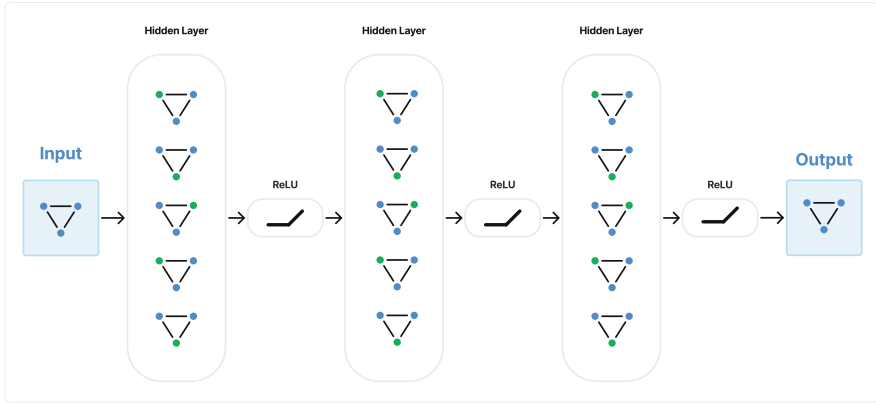


Fig. 1. Graph Representation of Network Data.

However, traditional GNNs sometimes overlook important edge attributes and struggle with heterogeneous graphs that integrate multiple types of system entities and operations.

To overcome these limitations, we propose an anomaly detection system that combines the strengths of the Activity and Event Network (AEN) model [2], Graph Neural Networks (GNNs), and Graph Attention Networks (GATs). Our approach introduces an attention mechanism where the weight assigned to a neighbor is based on its in-degree, reflecting its significance within the network. This allows the system to capture important structural and relational information better.

Our proposed method involves constructing a graph from network traffic data, applying the new attention mechanism within the GAT framework [3], and using this model to detect anomalies. By focusing on nodes with higher in-degree, our system prioritizes critical nodes and their interactions, improving detection accuracy.

In this paper, we detail our approach, including the graph construction process using the AEN model, the implementation of the attention-based GAT model, and the evaluation of its performance using real-world datasets.

The remainder of this paper is organized as follows. Section 2 reviews related works. Section 3 provides an overview of the AEN framework and its operational dynamics. Section 4 presents the methodology, including data collection, graph construction, and model implementation. Section 5 discusses the experimental setup and results. Section 6 concludes the paper and outlines future work.

2 Related Work

Many researchers have worked on detecting unusual activities in network traffic by applying machine learning and deep learning methods. In this section, we look at important studies in this field. We describe their approaches, the datasets they used, and the results they achieved. We also point out their limitations, which our new method aims to overcome.

Kisanga et al. [4] proposed a supervised Graph Convolutional Network (GCN)-based model to detect network anomalies using the Activity and Event Network (AEN) framework. In their approach, the authors constructed the network as a graph where each node represents a unique network entity and each edge represents an interaction or event between these entities. The proposed method involves extracting snapshots of the graph at specific time points to capture the state of the network, clustering these snapshots into smaller groups based on common nodes or edges, matching clusters by comparing the number of common nodes or edges between them, and measuring the difference between clusters using Graph Edit Distance (GED), which quantifies the cost of transforming one graph into another.

The GCN model used in this study consists of two layers, with the input layer taking node features and embeddings, and the output layer producing a probability score indicating whether a network event is normal or anomalous. The model incorporates 16 hidden layers with Rectified Linear Unit (ReLU) activation functions to introduce non-linearity and improve training speed. A log softmax function is used in the final layer to output probabilities. Their proposed GCN model achieved an accuracy score of 76% overall.

Zheng et al. [5] proposed a framework called AddGraph, designed for detecting anomalies in dynamic graphs using an attention-based temporal Graph Convolutional Network (GCN). In their approach, the authors represented the dynamic graph as a sequence of snapshots, with each snapshot capturing the state of the graph at a specific time point. They used a GCN to process the structural and content features of the nodes within each snapshot. To capture short-term patterns, a contextual attention-based model was applied to a local window of snapshots. The long-term hidden states obtained from the GCN were combined with the short-term patterns using Gated Recurrent Units (GRUs) to generate the final node representations.

The anomalous score for each edge (i, j, w) was computed using the following function:

$$f(i, j, w) = w \cdot \sigma(\beta \cdot (\|a \odot h_i + b \odot h_j\|_2^2 - \mu)) \quad (1)$$

where h_i and h_j are the hidden states of nodes i and j , respectively, $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, and a and b are parameters to optimize. The parameters β and μ are hyperparameters in the score function.

To handle the insufficient labeled anomaly data, selective negative sampling, and margin loss were introduced during training. The loss function used was:

$$L = \sum_{(i,j,w) \in E_t} \sum_{(i',j',w) \notin E_t} \max\{0, \gamma + f(i, j, w) - f(i', j', w)\} + \lambda L_{\text{reg}} \quad (2)$$

where γ is the margin between the possibilities of normal and anomalous edges, and L_{reg} is an L_2 -regularization term to avoid overfitting.

The AddGraph model was evaluated using two real-world datasets, the UCI Message dataset and the Digg dataset. The AddGraph model achieved AUC scores of 0.8083 and 0.8341 for 1% anomaly proportion on the UCI Message and Digg datasets, respectively.

Pujol-Perich et al. [6] proposed a Graph Neural Network (GNN) model for Network Intrusion Detection Systems (NIDS), focusing on capturing structural patterns of network traffic to improve robustness against adversarial attacks. In their approach, the authors represented network traffic as host-connection graphs, where each node corresponds to a host or a flow, and edges represent connections between hosts and flows. The proposed GNN model processes these graphs using a message-passing architecture designed to handle heterogeneous elements and the specific needs of network intrusion detection. The message-passing phase involves computing messages between connected nodes, aggregating these messages, and updating node states based on the aggregated information. The update function is approximated by neural networks, specifically using 2-layer fully connected networks for message functions and Gated Recurrent Units (GRUs) for state updates. The model was evaluated on the CIC-IDS2017 dataset. The GNN model achieved an accuracy range of 73% to 99% depending on the type of attacks.

Atkinson et al. [7] proposed an autoencoder-based strategy for anomaly detection in boosted jets, utilizing Graph Neural Networks (GNNs) to enhance detection performance. In their approach, the authors represented boosted jets as graphs where nodes correspond to jet constituents (microjets) and edges represent the connections between them. Each microjet's feature vector included observables such as transverse momentum ($\log p_T$), pseudorapidity ($\Delta\eta$), azimuthal angle ($\Delta\phi$), angular distance (ΔR), and normalized mass ($\bar{m}$). Edges were characterized by features like the distance between nodes using the Cambridge-Aachen metric (d_{CA}), k_T metric ($\log d_{kT}$), and invariant mass ($\log m_{ij}$). The GNN architecture employed in this study featured a message-passing phase and a graph readout layer. The message-passing function calculated messages between connected nodes, which were then aggregated to update node states. The encoder part of the autoencoder used NNConv and EdgeConv layers to process multidimensional edge and node features, while the decoder used a shared block of edge convolutions followed by specific edge reconstruction blocks.

Cai et al. [8] proposed a framework called StrGNN for detecting anomalies in dynamic graphs. In their approach, the authors constructed the network as a dynamic graph, where each node represents a system entity (such as a process or file) and each edge represents an interaction between these entities. The StrGNN framework consists of three key components, Enclosing Subgraph Generation (ESG), Graph Structural Feature Extraction (GSFE), and Temporal Detection Network (TDN). The ESG module generates h-hop enclosing subgraphs centered on the target edge from each graph snapshot. A node labeling function is proposed to identify the role of each node in the subgraph. The GSFE module leverages a Graph Convolutional Network (GCN) and pooling technologies to extract fixed-size features from each subgraph. The TDN employs Gated Recurrent Units (GRUs) to capture temporal dependencies for anomaly detection. The StrGNN model was evaluated on six benchmark datasets and a real enterprise security system. The benchmark datasets include UCI Messages, Digg, Email, Topology, Bitcoin-Alpha, and Bitcoin-OTC. The AUC scores were 0.8252 for 5% anomalies in the UCI Messages dataset and 0.8667 for 5% anomalies in the Bitcoin-Alpha dataset.

These works provide a solid foundation for the development of more advanced GNN models for anomaly detection. However, they often focus on static or semi-static graphs and may not fully capture the dynamic and evolving nature of network data. Our proposed attention-based GNN anomaly detection system aims to address these limitations by incorporating attention mechanisms and dynamic graph representations to improve detection accuracy and scalability.

3 Overview of the Activity and Event Network (AEN) Framework

This section explains how the AEN graph is built, with a focus on the key elements and important features of the new proposed method.

The Activity and Event Network (AEN) framework [2] is designed to monitor and analyze the security of networked systems in real time. This framework identifies and traces long-term and stealthy cyber-attacks that traditional methods might overlook. The core elements of the AEN framework include nodes representing entities and edges denoting their relationships, encapsulated in a dynamic uncertain directed multigraph structure. The AEN framework leverages a dynamic uncertain directed multigraph defined by the following:

$$G = (N, E, s, t, \Sigma_N, \Sigma_E, \ell_N, \ell_E, T, \rho_N, \rho_E, \rho_{\ell_N}, \rho_{\ell_E}) \quad (3)$$

where:

- N is the set of nodes.
- E is the set of edges.
- $s : E \rightarrow N$ maps edges to source nodes.
- $t : E \rightarrow N$ maps edges to target nodes.
- Σ_N is a set of node labels.

- Σ_E is a set of edge labels.
- $\ell_N : N \rightarrow \Sigma_N$ labels nodes.
- $\ell_E : E \rightarrow \Sigma_E$ labels edges.
- $T \subseteq \mathbb{R}^+$ is the time domain.
- $\rho_N : N \times T \rightarrow (0, 1]$ assigns existence probabilities to nodes over time.
- $\rho_E : E \times T \rightarrow (0, 1]$ assigns existence probabilities to edges over time.
- $\rho_{\ell_N} : \ell_N \times T \rightarrow (0, 1]$ assigns probabilities to node labels over time.
- $\rho_{\ell_E} : \ell_E \times T \rightarrow (0, 1]$ assigns probabilities to edge labels over time.

The process of constructing the AEN involves navigating through network traffic to generate edges and nodes using an edge creation function and a node creation function, respectively. The exact algorithms for these functions are detailed in [2]. However, the following two algorithms, Algorithm 1 and 2, demonstrate a high-level view of the process.

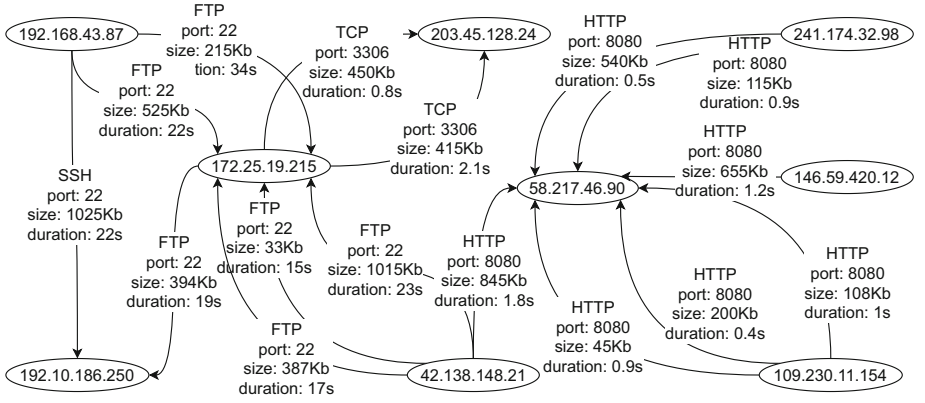


Fig. 2. Illustration of an AEN graph

This paper introduces an attention-based anomaly detection system that uses the AEN model to capture and continuously update the network's status. Our proposed anomaly detection system utilizes the alert node within the AEN to monitor for potential anomalies. A visual representation of an example AEN graph is presented in Fig. 2, where nodes represented by IP addresses are interconnected by edges that capture the flow of data, characterized by protocol type, port number, transfer size, session duration, etc. Additionally, Algorithm 3 illustrates the process that occurs when an alert is generated.

4 Proposed Anomaly Detection System

The proposed anomaly detection system leverages the strengths of Attention Graph Networks (GATs) [3] and the Activity and Event Network (AEN) model

Algorithm 1. Find or Insert Host Node**Require:** Graph G , IP address ip **Ensure:** Host node h

```

1:  $h \leftarrow \text{FindHost}(G, ip)$ 
2: if  $h = \emptyset$  then
3:    $h \leftarrow \text{newNode}()$ 
4:    $h.type \leftarrow \text{"HOST"}$ 
5:    $h.ip \leftarrow ip$ 
6:    $\text{InsertNode}(G, h)$ 
7:    $\text{AddIPInfo}(G, h)$ 
8: end if
9: return  $h$ 

```

Algorithm 2. Receive Packet**Require:** Graph G , packet p , time window δ **Ensure:** Updated graph G

```

1:  $source \leftarrow \text{FindOrInsertHost}(G, p.sourceIP)$ 
2:  $dest \leftarrow \text{FindOrInsertHost}(G, p.destIP)$ 
3:  $S \leftarrow \text{FindAllEdgesBetween}(G, \text{"SESSION"}, source, dest)$ 
4: if  $\exists s \in S$  such that  $\text{PacketMatchesSession}(s, p, \delta)$  then
5:    $s \leftarrow \text{AddPacketToSession}(s, p)$ 
6: else
7:    $s \leftarrow \text{CreateSession}(p, source, dest)$ 
8: end if
9:  $\text{SaveEdge}(G, s)$ 

```

Algorithm 3. Receive Alert**Require:** Graph G , raw alert $rawAlert$, time window δ **Ensure:** Updated graph G

```

1:  $A \leftarrow \text{FindAllNodes}(G, \text{"ALERT"})$ 
2: if  $\exists a \in A$  such that  $\text{MatchAlert}(a, rawAlert, \delta)$  then
3:    $a \leftarrow \text{UpdateAlert}(a, rawAlert)$ 
4: else
5:    $host \leftarrow \text{FindOrInsertHost}(G, alert.sourceIP)$ 
6:    $a \leftarrow \text{CreateAlert}(rawAlert)$ 
7:   Create and save edge between  $a$  and  $host$ 
8: end if
9:  $\text{SaveNode}(G, a)$ 

```

[2]. This combination is chosen for its alignment with real-time processing requirements and its capability to effectively manage event-based data.

The process initiates with the construction of the AEN graph from network traffic data.

4.1 Graph Attention Networks (GAT)

Graph Attention Networks (GATs) are a type of Graph Neural Network (GNN) that incorporates attention mechanisms to improve the learning of node representations. GATs assign different importance to different nodes within a neighborhood, which enhances the network's ability to focus on the most relevant parts of the graph.

In GATs, the attention mechanism is applied to the features of the neighboring nodes. Given a node i and its neighbor j , the attention coefficient α_{ij} is computed as follows:

$$e_{ij} = \text{LeakyReLU}(\mathbf{a}^T [\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_j]) \quad (4)$$

where: - $\mathbf{h}_i$ and $\mathbf{h}_j$ are the input features of nodes i and j . - $\mathbf{W}$ is a weight matrix. - $\parallel$ denotes concatenation. - $\mathbf{a}$ is a learnable weight vector. - LeakyReLU is the activation function applied to the concatenated features.

The attention coefficients are then normalized across all neighbors of the node i using the softmax function:

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})} \quad (5)$$

The normalized attention coefficients α_{ij} are used to compute a linear combination of the neighbors' features, resulting in the updated node features:

$$\mathbf{h}'_i = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W}\mathbf{h}_j \right) \quad (6)$$

where σ is a non-linear activation function such as ReLU.

The final node representations are obtained after L layers of GAT:

$$\mathbf{H}^{(L)} = \text{GAT}(\mathbf{H}^{(L-1)}, \mathbf{A}) \quad (7)$$

where $\mathbf{A}$ represents the adjacency matrix of the graph.

Figure 3 illustrates the structure of the Graph Attention Network GNN model. The model starts with an adjacency matrix representing the connections between nodes. Each node's features are then transformed using a learnable weight matrix. The attention mechanism computes the attention coefficients e_{ij} using Leaky ReLU and softmax functions to normalize these coefficients into α_{ij} . The final embeddings h' for each node are derived by aggregating the weighted sum of its neighbors' features.

4.2 Modified Attention Mechanism with In-Degree

In this work, we propose a modification to the standard attention mechanism by incorporating the in-degree of the neighboring nodes. The modified attention coefficient α_{ij} is computed as follows:

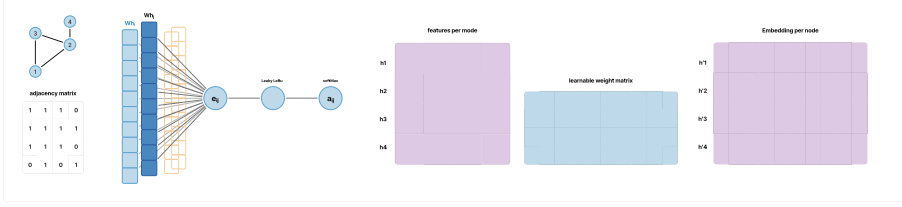


Fig. 3. Structure of the GAT model from input adjacency matrix to final node embeddings.

$$e_{ij} = \text{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_j] + \log(\deg(j) + 1)) \quad (8)$$

where $\deg(j)$ is the in-degree of node j . This modification allows the model to assign higher importance to neighbors with higher in-degrees, reflecting their potential significance in the network.

4.3 Training

The algorithm for training the model is summarized in Algorithm 4.

Algorithm 4. Training Model

Require: AEN Graph $G = (V, E)$ with node features $\mathbf{H}$, weight matrix $\mathbf{W}$, attention vector $\mathbf{a}$, learning rate η , number of epochs T

Ensure: Trained model parameters $\mathbf{W}$ and $\mathbf{a}$

- 1: Initialize $\mathbf{W}$ and $\mathbf{a}$
 - 2: **for** each epoch $t = 1$ to T **do**
 - 3: **for** each node $i \in V$ **do**
 - 4: **for** each neighbor $j \in \mathcal{N}_i$ **do**
 - 5: Compute e_{ij} using Equation (4)
 - 6: **end for**
 - 7: Compute α_{ij} using softmax function in Equation (2)
 - 8: Update node feature $\mathbf{h}'_i$ using Equation (3)
 - 9: **end for**
 - 10: Update $\mathbf{W}$ and $\mathbf{a}$ using gradient descent with learning rate η
 - 11: **end for**
 - 12:
 - 13: **return** $\mathbf{W}$ and $\mathbf{a}$
-

We train the GAT model using a supervised learning approach with labeled data. The loss function used is the cross-entropy loss:

$$\mathcal{L} = - \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (9)$$

where y_i is the true label and $\hat{y}_i$ is the predicted probability for node i .

The model is trained using the Adam optimizer with an initial learning rate of 0.01. We perform early stopping based on the validation loss to prevent overfitting.

The model is implemented using the PyTorch Geometric library [9]. The network data is preprocessed to extract node and edge features. The model architecture includes 4 layers of GAT, each with 128 hidden units. The dropout rate is set to 0.5 to prevent overfitting.

5 Performance Evaluation

To evaluate the performance of our proposed anomaly detection system, we used two datasets: the CICIDS2017 dataset [10] and the Tor-nonTor dataset [11]. Both datasets provide a comprehensive set of network traffic data, which is essential for testing the effectiveness of our model in detecting anomalies.

5.1 Datasets

CICIDS2017 Dataset. The CICIDS2017 dataset is a widely used benchmark for evaluating intrusion detection systems (IDS) and intrusion prevention systems (IPS). It includes a comprehensive set of benign and attack traffic, resembling real-world network data. The dataset contains both packet capture (PCAP) files and labeled flows (CSV files) generated using CICFlowMeter. The features include timestamps, source and destination IPs, ports, protocols, and attack labels. We utilized a subset of its DDoS dataset, specifically the Friday Afternoon DDoS Dataset. This subset includes 97,718 benign entries and 128,027 DDoS entries.

Tor-nonTor Dataset. The Tor-nonTor dataset (ISCXTor2016) is designed to distinguish between Tor and non-Tor traffic. It includes several traffic types such as web browsing, email, chat, audio streaming, video streaming, FTP, VoIP, and P2P. This dataset helps evaluate our model's ability to detect anomalies related to Tor network usage.

The dataset was created by capturing traffic using Wireshark and tcpdump, generating a total of 22 GB of data. We used scenario-A subset which contains 69,686 nonTor entries and 14,508 Tor entries.

5.2 Evaluation Metrics

We used several standard metrics to evaluate the performance of our anomaly detection system, including False Positive Rate (FPR), False Discovery Rate (FDR), Negative Predictive Value (NPV), F1 score, Accuracy (ACC), Specificity (SPC), Precision (PPV), Sensitivity (TPR), and False Negative Rate (FNR). These metrics provide a comprehensive view of the model's ability to correctly identify anomalies and minimize false positives and false negatives. Table 1 shows the definitions of the considered metrics.

Table 1. Considered Performance Metrics

Measure	Definition
False positive rate	$FPR = FP / (FP + TN)$
False discovery rate	$FDR = FP / (FP + TP)$
Negative predictive value	$NPV = TN / (TN + FN)$
F1 score	$F1 = (2 * TP) / (2TP + FP + FN)$
Accuracy	$ACC = (TP + TN) / (P + N)$
Specificity	$SPC = TN / (FP + TN)$
Precision	$PPV = TP / (TP + FP)$
Sensitivity	$TPR = TP / (TP + FN)$
False negative rate	$FNR = FN / (FN + TP)$

5.3 Results

To provide a clearer understanding of the model’s performance, we present the confusion matrices for both datasets in Table 2 and 3.

Table 4 shows the performance of our proposed model on the CICIDS2017 and Tor-nonTor datasets.

Table 2. Confusion Matrix for CICIDS2017 Dataset

	Predicted Positive	Predicted Negative
Actual Positive	112,784	15,243
Actual Negative	11,073	86,645

Table 3. Confusion Matrix for Tor-nonTor Dataset

	Predicted Positive	Predicted Negative
Actual Positive	12,254	2254
Actual Negative	5,556	64,130

The results from both datasets show that our proposed anomaly detection system is effective in identifying network anomalies. The high precision and recall values indicate that the model is capable of accurately detecting both known and novel anomalies, while the balanced F1-score suggests that it maintains a good trade-off between precision and recall.

The use of the attention mechanism in Graph Attention Networks (GAT) allows the model to focus on important nodes and their relationships, improving

Table 4. Performance Results

Metric	CICIDS2017	Tor-nonTor
False positive rate (FPR)	11.3%	7.9%
False discovery rate (FDR)	8%	31.2%
Negative predictive value (NPV)	85%	96.6%
F1 score	89.5%	75.8%
Accuracy	88.3%	90.7%
Specificity (SPC)	88.6%	92.0%
Precision (PPV)	91%	68.8%
Sensitivity (TPR)	88%	84.4%
False negative rate (FNR)	11.9%	15.5%

detection accuracy. The incorporation of the in-degree of nodes as part of the attention mechanism further enhances the model's ability to identify critical nodes in the network.

6 Conclusion

In this paper, we proposed an anomaly detection system using the Activity and Event Network (AEN) model and Graph Attention Networks (GAT). Our method gives more attention to neighbors with higher in-degree. This highlights their importance in the network. We combined the dynamic nature of AEN with the attention mechanisms of GAT. This allowed us to better identify anomalies in network behavior over time. We evaluated our approach using two datasets. The results showed that our system achieved an accuracy of 88.3% and 90.7%. Future work could explore the integration of additional features and further optimization of the attention mechanisms to enhance detection accuracy and scalability.

References

1. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **32**(1), 4–24 (2020)
2. Quinan, P.G., Traoré, I., Woungang, I.: Activity and event network graph and application to cyber-physical security. In: *Artificial Intelligence for Cyber-Physical Systems Hardening*, pp. 217–233. Springer, Cham (2022)
3. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. *stat* **1050**(20), 10–48550 (2017)
4. Kisanga, P., Woungang, I., Traore, I., Carvalho, G.H.: Network anomaly detection using a graph neural network. In: *2023 International Conference on Computing, Networking and Communications (ICNC)*, pp. 61–65. IEEE, 20 February 2023

5. Zheng, L., Li, Z., Li, J., Li, Z., Gao, J.: AddGraph: anomaly detection in dynamic graph using attention-based temporal GCN. In: IJCAI 2019, vol. 3, p. 7, 10 August 2019
6. Pujol-Perich, D., Suárez-Varela, J., Cabellos-Aparicio, A., Barlet-Ros, P.: Unveiling the potential of graph neural networks for robust intrusion detection. *ACM SIGMETRICS Perform. Eval. Rev.* **49**(4), 111–117 (2022)
7. Atkinson, O., Bhardwaj, A., Englert, C., Ngairangbam, V.S., Spannowsky, M.: Anomaly detection with convolutional graph neural networks. *J. High Energy Phys.* **2021**(8), 1–9 (2021)
8. Cai, L., et al.: Structural temporal graph neural networks for anomaly detection in dynamic graphs. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 3747–3756, 26 October 2021
9. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. arXiv preprint [arXiv:1903.02428](https://arxiv.org/abs/1903.02428) (2019 Mar 6)
10. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: *4th International Conference on Information Systems Security and Privacy (ICISSP)*, Portugal, pp. 201–211, January 2018
11. Lashkari, A.H., Draper-Gil, G., Mamun, M.S.I., Ghorbani, A.A.: Characterization of Tor traffic using time-based features. In: *Proceedings of the 3rd International Conference on Information System Security and Privacy, SCITEPRESS*, Porto, Portugal (2017)



Truckchain: A Blockchain-Powered IoT Real-Time Tracking System for Fuel Supply Chain Management

Cansa Kayembe Amaury¹ , Kabemba Ntumbwa Jonathan¹ ,
and Mpyana Mwamba Merlec² 

¹ Université Nouveaux Horizons, Lubumbashi, Democratic Republic of the Congo
jonathankabemba4100@gmail.com

² Korea University, Seoul, South Korea
mlecjm@korea.ac.kr

Abstract. The fuel supply chain is a vital infrastructure demanding real-time traceability to ensure integrity, transparency, and security of the delivery. Traditional fuel supply chains are often plagued by complexities and vulnerabilities, including fraud, theft, and operational inefficiencies that require innovative solutions. This paper proposes Truckchain, a blockchain-powered Internet of Things (IoT) system designed to overcome these challenges and improve fuel supply chain traceability, transparency, security, and efficiency. Using Swala Logistics as a case study, Truckchain integrates IoT sensors with blockchain technology to create an immutable, decentralized, and transparent real-time tracking system that monitors fuel from loading to delivery. The system captures critical data, such as temperature, fuel levels, and delivery quantities, and securely stores them on a blockchain to ensure the integrity and immutability of the data. Truckchain enhances stakeholder trust through complete transparency, improves regulatory compliance, and facilitates effective anomaly detection and loss prevention by providing comprehensive real-time monitoring functionality. This paper details the architecture, design, implementation, and evaluation of Truckchain, highlighting its potential to enhance fuel supply chain efficiency and security significantly.

Keywords: Blockchain · Internet of Things · Supply Chain Management · Real-Time Tracking System · Fuel Supply Chain · Smart Contract

1 Introduction

Fuel supply chains are critical to the global economy, supporting key industries such as transportation, manufacturing, and energy production, which depend on reliable, secure, and efficient fuel delivery [1–4]. However, these supply chains

Supported by Université Nouveaux Horizons (UNH) and Korea University funds.

face significant challenges, including theft, fraud, and operational inefficiencies, which can result in substantial economic losses, environmental risks, and service disruptions [5–8]. Traditional fuel supply chain management systems often fall short, lacking the real-time visibility, security, and efficiency needed to effectively address these issues [7–10]. These limitations have led to an increase in interest in the use of advanced technologies to address the unique challenges faced by fuel supply chains [11–14].

Recently, blockchain and IoT (Internet of Things) technologies have emerged as promising solutions in supply chain management [11–15]. Blockchain technology [16, 17], with its decentralized and immutable ledger, enables secure and transparent tracking of transactions and asset movements throughout the supply chain, providing an indelible record that can deter fraud and enhance accountability [11–15]. Meanwhile, IoT devices facilitate real-time monitoring of fuel assets, collecting critical data on factors such as location, temperature, fuel levels, and quantities, which are essential to maintain fuel quality and ensure delivery integrity [12–15, 18, 19, 21]. A key enabler of blockchain functionality is its consensus mechanism, which ensures data integrity and agreement across decentralized nodes in a peer-to-peer network [16, 17, 19]. Consensus algorithms, such as Proof of Work (PoW) and Proof of Authority (PoA), validate transactions and prevent tampering, making the data trustworthy for all stakeholders [17, 20]. This ensures that every recorded transaction within the fuel supply chain—whether related to transportation, storage, or delivery—is accurate and verifiable. Adding to these capabilities, smart contracts—a key feature of blockchain—play a key role in automating processes within the fuel supply chain [18, 22].

Smart contracts are self-executing programs that enforce the terms of an agreement without the need for intermediaries, ensuring secure and automated operations [17, 18, 22]. In the context of fuel logistics, smart contracts can automatically verify delivery data, release payments, and trigger alerts for anomalies, reducing manual interventions and operational inefficiencies. Together, these technologies offer an integrated framework that improves transparency, security, and operational efficiency across supply chains [23–26]. This integration of blockchain, IoT, and smart contracts provides a cohesive system to streamline complex logistics workflows while ensuring data accuracy and traceability.

Despite the growing adoption of blockchain and IoT in various supply chains, including agriculture, food, pharmaceuticals, and automotive industries, research focusing on fuel supply chains remains limited [10–13, 27–32]. This gap is particularly pronounced in developing regions, where challenges such as real-time tracking, regulatory compliance, and theft prevention remain inadequately addressed. The absence of scalable, practical solutions tailored to the unique requirements of fuel logistics in remote and high-risk environments underscores the need for further research and innovation in this domain.

1.1 Contribution

To address this gap, this paper proposes Truckchain, a blockchain-powered IoT real-time tracking system designed to enhance transparency, security, and oper-

ational efficiency across the fuel supply chain. This paper discusses the design, implementation, and evaluation of the proposed solution developed to enhance the fuel supply chain management for Swala Logistics, a transport company operating in the Democratic Republic of Congo (DRC)¹. Swala Logistics serves gas stations and mining sites across the Haut-Katanga and Lualaba provinces, where challenges in maintaining fuel security and delivery reliability are particularly acute. By integrating blockchain, smart contracts, and IoT technologies, Truckchain provides real-time traceability through IoT sensors, enhances regulatory compliance, and improves transparency and efficiency in fuel logistics operations. Truckchain integrates key features such as immutable blockchain records to prevent fraud and automated smart contracts for transaction validation, payment releases, and anomaly detection. The system ensures compliance with regulatory requirements, detects irregularities such as unauthorized fuel withdrawal, and provides user-friendly dashboards for real-time monitoring and analytics. This paper aims to demonstrate Truckchain's potential to modernize fuel supply chains while addressing critical vulnerabilities associated with traditional systems.

1.2 Methodology

This study employs a hybrid approach that combines quantitative and qualitative research methods to evaluate the potential of the Truckchain system to improve fuel supply chain management in Swala Logistics, as described below.

- *Literature Review*: A comprehensive literature review was conducted to identify the existing research gap on blockchain in fuel supply chain management. This review provides a theoretical foundation for the design and implementation of the prototype of the proposed Truckchain system.
- *Case Study*: An in-depth case study of Swala Logistics was conducted to identify specific challenges and opportunities for improvement in the company's fuel supply chain. This case study involved interviews with key stakeholders and an analysis and evaluation of the existing fuel supply chain management system, strategies, and practices.
- *System Design and Implementation*: Based on the literature review and case study findings, a detailed architecture for the Truckchain system was designed and a prototype was developed to demonstrate its feasibility. This architecture integrates IoT sensors, smart contracts, and blockchain technology to provide real-time traceability and data integrity. The prototype was built using appropriate data collection, processing, and storage technologies, including Ethereum blockchain and LoRaWAN communication protocols.
- *Evaluation*: The performance of the Truckchain system was evaluated against predefined objectives, including traceability accuracy, cost reduction, and operational efficiency. This evaluation involved testing the system in a real-world environment and assessing its operational costs and impact on the efficiency and security of the Swala Logistics fuel supply chain.

¹ <https://www.linkedin.com/company/swala-logistics/> (Accessed on 28.08.2024).

The rest of this study is organized as follows. Section 2 presents a review of the literature on the application of blockchain and IoT in supply chains. Section 3 describes the design and architecture of the system. Section 4 covers the implementation and evaluation of the prototype of the proposed system. Finally, Sect. 5 concludes this study by discussing future research directions.

2 Literature Review

Supply chain management faces critical challenges, including the need for traceability, transparency, and security, which are essential for operational efficiency and stakeholder trust [1–6]. Traditional systems often lack the real-time visibility and secure data-sharing capabilities necessary to mitigate risks such as fraud, theft, and inefficiencies [7–10]. As a result, the sector is increasingly adopting advanced technologies such as blockchain and IoT to address these limitations by offering improved data integrity, enhanced security, and better coordination among stakeholders [11–15].

2.1 Challenges in Current Fuel Supply Chain Management

Fuel supply chains are complex and face unique challenges compared to other industries [2–4]. These include ensuring delivery integrity, preventing theft, reducing fraud, and maintaining compliance with regulatory standards [3–10]. Traditional fuel logistics systems rely heavily on manual processes, which are prone to errors, inefficiencies, and tampering [4–6]. Additionally, the lack of real-time visibility across the supply chain often results in delays, mismanagement, and increased operational costs [5–7]. Another major concern is the lack of robust traceability mechanisms to monitor critical parameters such as fuel quality, volume, and temperature during transportation [6–8]. These issues compromise the security and efficiency of the supply chain, leaving it vulnerable to fraud and operational disruptions [8–11]. Addressing these challenges requires innovative solutions that enhance transparency, automate processes, and provide real-time insights into supply chain operations.

2.2 Blockchain and IoT in Supply Chains

Blockchain is a decentralized, distributed ledger technology that records transactions across multiple nodes in a secure and tamper-proof manner [12–20]. Its core features include immutability, transparency, and decentralization, which make it an ideal tool for ensuring data integrity and accountability. Blockchain eliminates the need for intermediaries by allowing stakeholders to independently verify and trust recorded data [13–23]. This makes it particularly valuable in industries such as supply chains requiring high traceability standards, where it reduces errors, prevents tampering, and clarifies responsibilities [24–26].

The Internet of Things (IoT) refers to a network of interconnected devices equipped with sensors and communication capabilities, enabling real-time data

collection and sharing [13,18,19]. IoT technology provides insights into critical supply chain parameters such as location, temperature, and condition of goods, allowing for proactive management and improved decision-making [15,21,23]. When combined with blockchain, IoT enhances the security and traceability of supply chains by ensuring that collected data is securely recorded and shared among stakeholders [24–31].

The integration of blockchain and IoT is redefining supply chain management practices by enabling real-time data collection and secure storage [24–26]. IoT devices monitor key parameters, while blockchain provides a transparent and tamper-proof platform for recording and verifying this data. For example, blockchain has been successfully implemented in agriculture and food supply chains to improve traceability, ensuring product safety and authenticity [27,28].

Similarly, blockchain and IoT integration in pharmaceutical supply chains enhance drug traceability, ensuring authenticity and preventing counterfeit medications—critical for patient safety and regulatory compliance [13,29,30]. In the automotive industry, these technologies have enabled efficient tracking of components and vehicles, improving inventory management and ensuring supply chain transparency [12,31–33].

The adoption of blockchain and IoT in supply chains also facilitates decentralized access control systems, allowing dynamic permission management without central oversight [18,22]. This capability is essential for managing complex logistics with real-time requirements [31–33]. Projections suggest that blockchain adoption in supply chain management could grow at an annual rate of up to 87%, driven by the demand for transparent, secure, and efficient traceability systems [24–26]. Together, blockchain and IoT provide a robust foundation for addressing the traceability and operational challenges in modern supply chains.

2.3 Issues with the Existing System: Case Study

Swala Logistics, a leading transport and logistics company in the DRC, faces significant challenges in managing its fuel supply chain. The company's existing fuel management system does not provide real-time data on the location, condition, or quantity of fuel in transit, leading to vulnerabilities such as theft and fraud. Furthermore, the lack of real-time monitoring and verification mechanisms complicates the accurate measurement and billing of fuel deliveries, particularly in remote areas with limited network coverage.

Swala Logistics' fuel delivery process follows several key stages, from loading to delivery, and meets international ISO standards². The company uses an Integrated Management System (IMS) [34], which unifies various management systems, such as quality (ISO 9001), environmental (ISO 14001), and safety (ISO 45001), into a cohesive framework to enhance efficiency, compliance, and

² ISO stands for International Organization for Standardization. It is a non-governmental organization that develops international standards for various industries and products. These standards guarantee quality, safety, efficiency, and compatibility across borders [35].

traceability in fuel supply chain management [34,35]. These certifications require rigorous management of quality, environment, and safety in its operations. However, despite the methods in place for optimal transactions, many operational inefficiencies in the supply chain are caused by the following factors:

- *Inability to track cargo directly*: There is no device to track the fuel, its condition, quantity, or any other important information.
- *Inability to track driver movements in real-time*: During transit, especially when not in convoy, the driver is the sole authority on board, making centralized information and immediate verification impossible.
- *Complicity during unloading*: Complicity can facilitate theft, as invoicing depends on arbitrarily recorded information.
- *Inability to ensure and prove the quantity of the loaded fuel*: Real-time tracking of fuel variations, notably due to temperature changes, remains an unresolved challenge.
- *Issues related to unloading at independent clients*: Independent clients impose their unloading methods, leading to calibration and standardization issues, making client honesty essential for correct billing.
- *Difficult network coverage*: Outside urban areas, obtaining information becomes problematic during certain fuel transportation phases.

To address these challenges, this paper explores the integration of IoT and blockchain technologies to improve transparency, security, and efficiency in fuel supply chain management systems. Unlike prior studies, which have focused on sectors such as food and pharmaceuticals, this study tackles challenges unique to fuel supply chains, such as theft prevention, real-time tracking in remote areas, and secure monitoring of fuel quality and quantity. By providing decentralized, immutable tracking and dynamic access control, Truckchain enhances security and efficiency, setting a new standard in fuel supply chain management and bridging gaps in existing research.

3 Truckchain System Design

3.1 System Architecture

Figure 1 depicts the Truckchain system architecture designed to provide real-time traceability and data integrity across the fuel supply chain. The architecture comprises several layers, each responsible for different aspects of the system's functionality:

- *Application Layer*: This layer provides user interfaces for various stakeholders including the Swala Logistics Back Office (SLBO) and Client Dashboard. These interfaces provide real-time data on truck locations, fuel quantities, and temperatures, ensuring transparency and visibility throughout delivery.
- *Blockchain Layer*: The blockchain layer uses Ethereum's decentralized ledger to record all data collected by IoT sensors securely. Smart contracts, executed on the Ethereum Virtual Machine (EVM) [16], automate transaction validation and ensure data immutability.

- *Communication Layer*: The communication layer facilitates data transmission between IoT sensors and the central server using the LoRaWAN protocol [36–38]. LoRaWAN was chosen for its long-range and low-power capabilities, which makes it ideal for tracking fuel trucks in remote areas.
- *Business Process Layer*: This layer encompasses the operational processes related to fuel loading and unloading. IoT sensors embedded in trucks collect data on fuel volume, temperature, and location, which is then transmitted in real-time to the blockchain for secure recording.

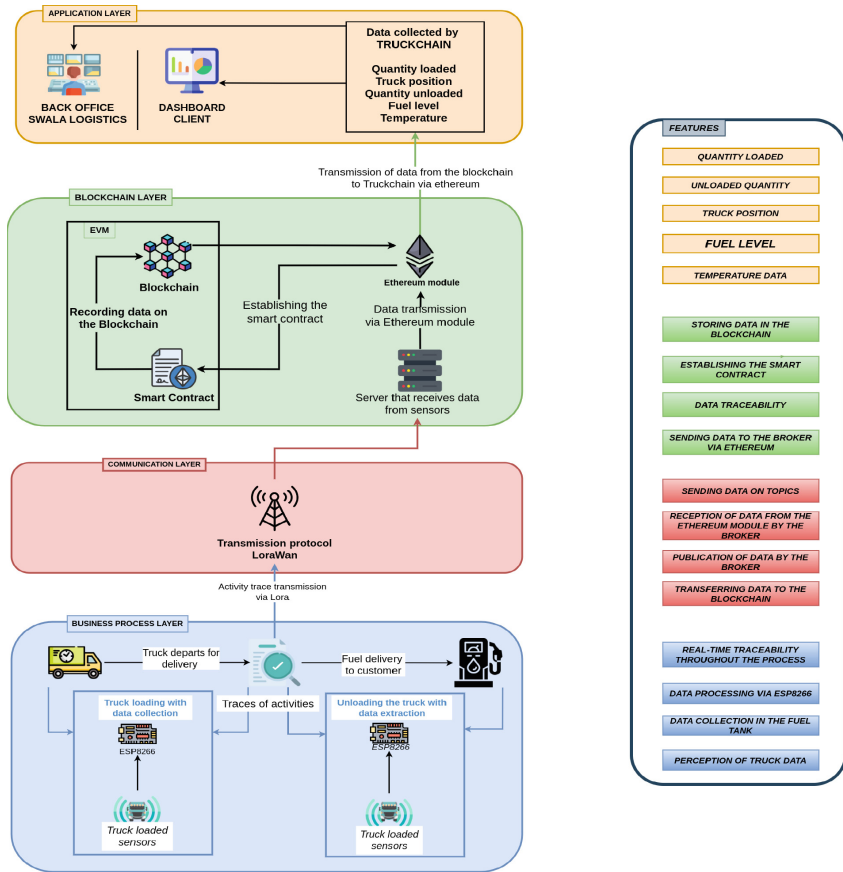


Fig. 1. Architecture diagram of the proposed Truckchain—a real-time tracking system for secure, transparent, and efficient fuel supply chain management.

This architecture (Fig. 1) provides real-time tracking and transparency for the fuel supply chain by integrating IoT sensors, data security via blockchain, and reliable information transmission through LoraWan, thereby ensuring the

integrity and reliability of the delivery process. It is supported by a comprehensive software design that underpins its functionality.

3.2 IoT Integration and Data Collection

IoT devices are embedded in fuel trucks to monitor various parameters, including fuel levels, temperature, and GPS location. These devices are equipped with sensors that continuously collect data, which is then transmitted to the central server via the LoRaWAN communication protocol [37,38]. Using LoRaWAN ensures reliable data transmission even in remote areas with limited network coverage. The collected data is processed and stored on the blockchain, ensuring that all transactions and movements are recorded in an immutable ledger. IoT and blockchain technology integration provides a robust solution for real-time monitoring and traceability of fuel shipments.

3.3 Data Model Diagram

Figure 2 illustrates the *Entity-Relationship Diagram (ERD)* that defines the data structure for managing the core entities: users, customers, drivers, trucks, orders, and stations. Each entity and its relationships are logically organized to ensure efficient data handling:

- *Users*: Includes two specific roles, Admin and Customer, with role-specific attributes.
- *Drivers and Trucks*: Drivers are linked to the trucks they operate, with each truck capable of carrying multiple orders.
- *Orders*: Each order is associated with a customer, a truck, and a delivery location, including detailed information about the order's status.
- *Stations*: Trucks can refuel at multiple stations, which store information such as location and available fuel types.

This data model ensures seamless interaction between entities and supports the efficient operation of the Truckchain system by providing a logical framework for data management and interaction.

3.4 Use Case

Figure 3 illustrates the main use cases of the Truckchain system, highlighting the distinct roles of the Customer and the Admin:

- *Customer Use Cases*: Customers can create and confirm their accounts, place orders, track their status, and verify order details.
- *Admin Use Cases*: The Admin has extended privileges, including user management, order creation, and truck registration. Admins can also consult historical data related to customers and orders, ensuring operational oversight.

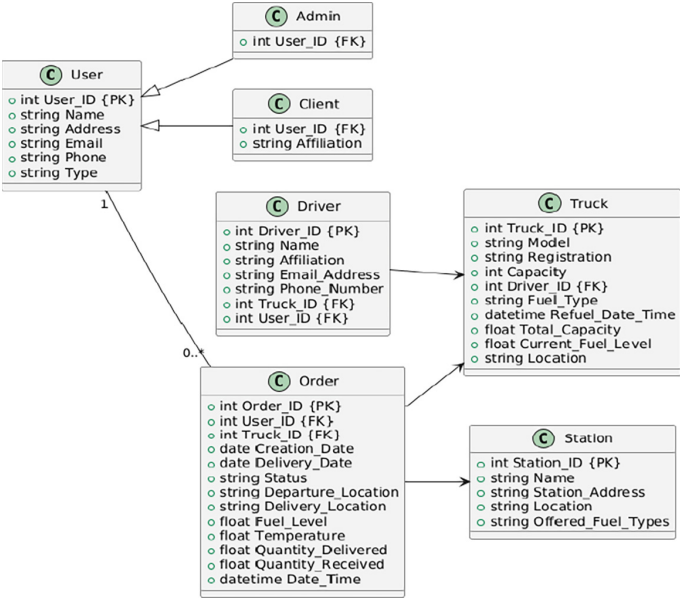


Fig. 2. Entity-Relationship Diagram (ERD) of the proposed Truckchain system.

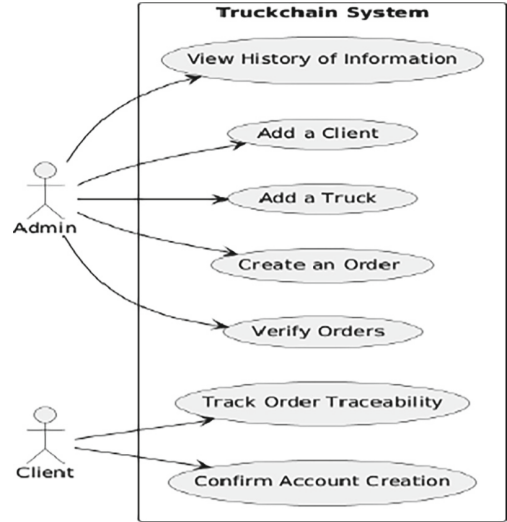


Fig. 3. Use case diagram of the proposed Truckchain system.

These use cases illustrate the system’s role-based access structure, allowing users to perform tasks specific to their responsibilities. This description of roles ensures efficient operations and prevents unauthorized access to sensitive data.

4 Implementation and Evaluation

Figure 4 shows a prototype of the Truckchain system that was developed to demonstrate its ability to track fuel shipments comprehensively, ensuring transparency and security throughout the supply chain. The system collects and processes critical data, including fuel level, truck location, tank temperature, delivery and receipt quantities, and tank status. This data enables real-time monitoring, accurate record-keeping, and informed decision-making for all stakeholders.

4.1 Implementation

Smart contracts are core components of the Truckchain system, automating key operations such as transaction validation and payment processing. Developed in Solidity and deployed on the Ethereum blockchain [16], these contracts ensure that all transactions are executed according to predefined rules and that the data recorded on the blockchain remains immutable and tamper-proof. To secure sensitive information, the blockchain implementation incorporates encryption and access control mechanisms, ensuring that only authorized stakeholders can access the data. The Truckchain system integrates IoT devices, blockchain technology, and custom software tools to deliver a robust solution for fuel supply chain management. The implementation process involved:

- *Programming IoT Sensors:* IoT sensors described in Table 1, programmed in C++, collect data such as fuel volume, temperature, and GPS location. This data is transmitted using the LoRaWAN protocol [36–38], enabling long-range communication for real-time monitoring and traceability³.
- *Developing Smart Contracts:* Smart contracts were developed in Solidity to automate the validation and recording of sensor data on the blockchain⁴. These contracts ensure data accuracy and facilitate automated payment releases based on predefined conditions.
- *Creating User Interfaces:* The SLBO and Client Dashboard were developed using Ruby on Rails and Node.js, with user-friendly interfaces allowing stakeholders to monitor fuel deliveries, track truck locations, and access historical data in real-time⁵.

³ <https://github.com/1Cansa/Truckchain/blob/master/truckSensor.cpp>.

⁴ <https://github.com/1Cansa/Truckchain/blob/master/truckSensor.sol>.

⁵ <https://github.com/1Cansa/Truckchain/tree/master/truckChainMain>.

Table 1. Experimental Prototype Hardware Setup.

Hardware (Computer and Sensors)	Utility
Computing Node	Dell Latitude 7840 with Intel Core i7 vPro, 8 GB RAM, and 256 GB SSD
Microcontroller ESP8266 (NodeMCU)	Integrated microcontroller with Wi-Fi connectivity for data processing and transmission
Flow Sensor YF-S201	Measures the liquid flow rate in the pipe for precise monitoring
Temperature Sensor DS18B20	Captures the temperature of the transported fluid for condition monitoring
GPS Module NEO-7M	Provides GPS coordinates for real-time vehicle location tracking
Ultrasonic Sensor HC-SR04	Measures the distance to the fluid surface to monitor tank levels
LoRaWAN SX1278 Module	Enables long-range, low-power communication with the base station

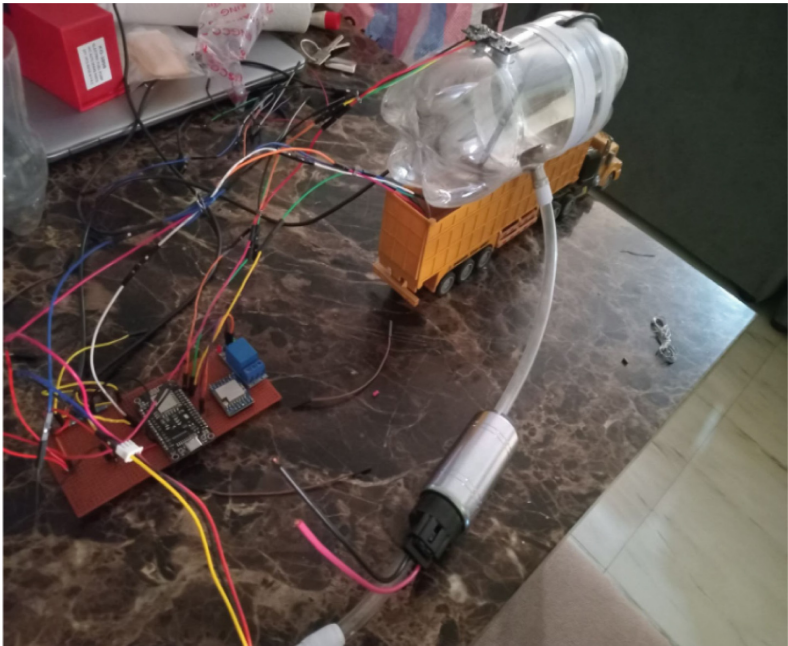


Fig. 4. Experimental testbed components and devices used for prototyping a fuel delivery tanker truck integrating sensors and applications for real-time tracking.

4.2 Hardware Specifications

A physical prototype of a fuel delivery tanker truck was built to simulate and validate the system’s functionalities. The prototype integrates a variety of com-

puting nodes and sensors as shown in Fig. 4, including a flow sensor, temperature sensor, GPS module, ultrasonic sensor, and an ESP8266 microcontroller, along with essential electronic components such as pumps and data transmission modules. This setup effectively demonstrates the real-time collection and secure transmission of data to the blockchain, ensuring data integrity and operational transparency. Table 1 provides an overview of the hardware components used in the prototype setup, highlighting their roles in enabling system functionality.

4.3 Experimental Evaluation

The experimental evaluation of the Truckchain system was conducted in real-world scenarios to assess its performance and its impact on the efficiency and security of Swala Logistics' fuel supply chain. Key performance indicators such as traceability accuracy, operational costs, and execution times were measured.

a. Smart Contract Evaluation: The smart contracts deployed on the Ethereum blockchain were evaluated for their efficiency and cost-effectiveness. A key metric in this evaluation was the transaction execution gas cost, which represents the computational effort required to execute operations on the Ethereum network [16]. Gas represents the computational effort required to execute operations on the Ethereum network, such as deploying smart contracts or executing functions.

The transaction execution gas cost is the total amount of gas consumed by an operation, multiplied by the gas price (set in Gwei, where $1 \text{ Gwei} = 10^{-9} \text{ ETH}$) [16]. Gas costs depend on factors such as the operation's complexity, network congestion, and the gas price set by the user. This evaluation calculated gas costs based on Ethereum pricing as of August 19, 2024 [39], demonstrating that while the initial deployment of smart contracts incurs some cost, the long-term benefits of enhanced security and reduced operational risks justify these expenses. Table 2 outlines the gas costs in Gwei with ETH and USD equivalent values associated with deploying the main smart contract of the proposed system. Table 3 outlines the operational costs for executing core functions of the Truckchain system. The results demonstrate the varying gas consumption required for different operations:

- *Write Functions:* Functions such as `setFlowRateSensor1`, `setTemperatureSensor1`, and `setUltrasonicLevel`, which involve updating the blockchain state, consumed higher gas values, ranging from 100,000 to 200,000 Gwei. This corresponds to costs of 0.262 to 0.524 USD per operation. These higher costs are expected due to the computational effort required to write new data to the blockchain securely and immutable.
- *Read/View Functions:* Functions such as `getFlowRateSensor1` and `getTemperatureSensor1`, which retrieve data from the blockchain without altering its state, consumed significantly less gas at 21,000 Gwei, translating to 0.0551 USD per query. These low costs highlight the efficiency of retrieving data in the Truckchain system.

- *Pure Functions*: The `abs` function, which does not interact with the blockchain state and operates independently, consumed the least gas at 15,000 Gwei, costing only 0.0393 USD. This demonstrates the minimal computational resources required for simple operations.

Table 2. Main smart contract deployment gas cost.

Smart Contract	Gas (Gwei)	ETH	USD [†]
<code>truckSensor.sol</code>	2,500,000	0.0025	6.5539
[†] ETH Price: 1 ETH = 2621.57 USD (2024.08.19) [39].			

Thus, the results show the system’s cost-efficiency, especially for frequent read and computation operations, making Truckchain suitable for real-time monitoring and data queries in cost-sensitive environments such as fuel supply chain management. The scalability of these costs ensures that even with increased transaction volumes, the system remains economically viable.

Table 3. Operational gas costs of core functions.

Function	Gas (Gwei)	ETH	USD [†]
<code>setFlowRateSensor1</code>	200,000	0.0002	0.524
<code>setFlowRateSensor2</code>	100,000	0.0001	0.262
<code>setTemperatureSensor1</code>	150,000	0.00015	0.393
<code>setUltrasonicLevel</code>	100,000	0.0001	0.262
<code>getFlowRateSensor1</code> (view)	21,000	0.000021	0.0551
<code>getFlowRateSensor2</code> (view)	21,000	0.000021	0.0551
<code>getTemperatureSensor1</code> (view)	21,000	0.000021	0.0551
<code>getUltrasonicLevel</code> (view)	21,000	0.000021	0.0551
<code>abs</code> (pure)	15,000	0.000015	0.0393

[†]ETH Price: 1 ETH = 2621.57 USD (2024.08.19) [39].

b. Efficiency Metrics Evaluation: The system’s efficiency was assessed through the execution times of smart contract functions under standard network conditions. While transaction times vary due to factors such as network congestion, gas price settings, and contract complexity, the system was optimized to minimize delays. Table 4 presents the execution times of core functions for the system. The result shows that the deployment of the smart contract required approximately 1.3s, which is expected given the complexity and initial setup

processes on the Ethereum blockchain. For operational functions such as setting sensor values (e.g., `setFlowRateSensor1` and `setTemperatureSensor1`), the execution times ranged between 0.045 and 0.06 s, indicating that the system can handle data updates promptly even under typical blockchain network conditions. View functions (e.g., `getFlowRateSensor1`) demonstrated excellent performance with execution times of approximately 0.01 s, benefiting from their lower computational requirements since they do not alter the blockchain state. The pure function *abs*, which operates independently of the blockchain state, performed even more efficiently, with an execution time of just 0.005 s, showcasing the system’s capability to process computations quickly.

Table 4. Execution time for main functions.

Action	Run times (seconds)
Contract deployment	1.3
<code>setFlowRateSensor1</code>	0.05
<code>setFlowRateSensor2</code>	0.045
<code>setTemperatureSensor1</code>	0.06
<code>setUltrasonicLevel</code>	0.055
<code>getFlowRateSensor1</code> (view)	0.01
<code>getFlowRateSensor2</code> (view)	0.01
<code>getTemperatureSensor1</code> (view)	0.01
<code>getUltrasonicLevel</code> (view)	0.01
<i>abs</i> (pure)	0.005

Thus, the results demonstrate that the Truckchain system achieves low-latency operation, ensuring critical updates and queries can be executed in near real-time. This makes it well-suited for dynamic environments such as fuel supply chain management.

c. System Performance Scalability: The Truckchain system is designed with scalability, enabling it to efficiently handle increasing volumes of transactions and data as the fuel supply chain expands. To evaluate its performance, the system was tested under varying conditions, including high transaction loads and diverse operational scenarios. The results demonstrated that Truckchain effectively scales with no significant degradation in performance, ensuring that system efficiency and reliability are preserved even as transaction volumes grow. This robustness highlights the system’s capacity to support the dynamic and evolving demands of large-scale fuel logistics operations.

5 Conclusion and Future Work

Truckchain represents a transformative solution for fuel supply chain management systems, integrating blockchain and IoT technologies to address the critical challenges of transparency, security, and operational efficiency. By providing real-time traceability and an immutable, decentralized tracking system, Truckchain ensures the integrity of fuel deliveries and monitors critical parameters such as temperature, fuel levels, and delivery quantities, which helps mitigate issues such as fraud, theft, and inefficiencies; thereby, enhancing the overall reliability of fuel logistics. The proposed system's design, based on the Swala Logistics case study, demonstrates its potential to foster stakeholder trust, improve regulatory compliance, and enhance anomaly detection capabilities. By providing a robust framework for secure and transparent fuel tracking, Truckchain positions itself as a vital solution for modernizing supply chain operations. However, challenges remain, including the complexity of transaction verification in public blockchain networks and managing the large volumes of data generated. Future work will focus on optimizing system performance, scalability, and reliability to address these challenges and meet the growing demands of the fuel industry. Further research could explore the integration of machine learning algorithms for predictive analytics to detect and prevent disruptions in the supply chain. Additionally, continuous updates will be necessary to ensure Truckchain adapts to evolving regulatory requirements and technological advancements, ensuring its sustained effectiveness in improving fuel supply chain operations.

Acknowledgment. This work was supported by the Université Nouveaux Horizons (UNH) researchfunding and Korea University Computer Science Brain Korea 21 (BK21) Four funding grant.

References

1. Mangan, J., Lalwani, C.: *Global Logistics and Supply Chain Management*. Wiley (2016)
2. Kaliani Sundram, V.P., Chandran, V.G.R., Awais Bhatti, M.: Supply chain practices and performance: the indirect effects of supply chain integration. *Benchmarking Int. J.* **23**(6), 1445–1471 (2016)
3. Martinez-Valencia, L., et al.: Supply chain configuration of sustainable aviation fuel: review, challenges, and pathways for including environmental and social benefits. *Renew. Sustain. Energy Rev.* **152**, 111680 (2021)
4. Wolff, M., Becker, T., Walther, G.: Long-term design and analysis of renewable fuel supply chains—an integrated approach considering seasonal resource availability. *Eur. J. Oper. Res.* **304**(2), 745–762 (2023)
5. Khalili, S.M., Pooya, A., Kazemi, M., Fakoor Saghih, A.M.: Integrated resilient and sustainable gasoline supply chain model with operational and disruption risks: a case study of Iran. *Environ. Dev. Sustain.*, 1–73 (2024)
6. Dabbene, F., Gay, P., Tortia, C.: Traceability issues in food supply chain management: a review. *Biosys. Eng.* **120**, 65–80 (2014)

7. Jafari, H., Ghaderi, H., Malik, M., Bernardes, E.: The effects of supply chain flexibility on customer responsiveness: the moderating role of innovation orientation. *Prod. Plann. Control* **34**(16), 1543–1561 (2023)
8. Couzineau-Zegwaard, E., Meier, O.: The evolution of the supply chain function within corporate governance through the lens of “organizational ambidexterity”. *Projectics/Proyectica/Projectique* **26**(2), 53–76 (2020)
9. Morgan, T.R., Gabler, C.B., Manhart, P.S.: Supply chain transparency: theoretical perspectives for future research. *Int. J. Logistics Manag.* **34**(5), 1422–1445 (2023)
10. Thakur, M., Hurburgh, C.R.: Framework for implementing traceability system in the bulk grain supply chain. *J. Food Eng.* **95**(4), 617–626 (2009)
11. Sheel, A., Nath, V.: Effect of blockchain technology adoption on supply chain adaptability, agility, alignment and performance. *Manag. Res. Rev.* **42**(12), 1353–1374 (2019)
12. Reimers, T., Leber, F., Lechner, U.: Integration of blockchain and the internet of things in an automotive supply chain. In: *IEEE International Conference on Decentralized Applications (DAPPCON)*, Newark, CA, USA, p. 15 (2019)
13. Rejeb, A., Keogh, J.G., Treiblmaier, H.: Leveraging the internet of things and blockchain technology in supply chain management. *Future Internet* **11**(7), 161 (2019)
14. Vincenzo, V., Antonello, C., Francesca, M., Mauro, C.: Sustainable supply chains with blockchain, IoT and RFID: a simulation on order management. *Sustainability* **13**(11), 6372 (2021)
15. Balamurugan, S., Ayyasamy, A., Joseph, K.S.: IoT-blockchain driven traceability techniques for improved safety measures in food supply chain. *Int. J. Inf. Technol.* **14**(2), 1087–1098 (2022)
16. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **151**(2014), 1–32 (2014)
17. Merlec, M.M., In, H.P.: Blockchain-based decentralized storage systems for sustainable data self-sovereignty: a comparative study. *Sustainability* **16**(17), 7671 (2024)
18. Ngwira, L., Merlec, M.M., Lee, Y.K., In, H.P.: Towards context-aware smart contracts for blockchain IoT systems. In: *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju Island, Republic of Korea, pp. 82–87. IEEE (2021)
19. Tsang, Y.P., Choy, K.L., Wu, C.H., Ho, G., Lam, H.Y.: Blockchain-driven IoT for food traceability with an integrated consensus mechanism. *IEEE Access* **7**, 129000–129017 (2019)
20. Islam, M.M., Merlec, M.M., In, H.P.: A comparative analysis of proof-of-authority consensus algorithms: aura vs clique. In: *2022 IEEE International Conference on Services Computing (SCC)*, Barcelona, Spain, pp. 327–332. IEEE (2022)
21. Aryal, A., et al.: The emerging big data analytics and IoT in supply chain management: a systematic review. *Supply Chain Manag. Int. J.* **25**(2), 141–156 (2020)
22. Merlec, M.M., In, H.P.: SC-CAAC: a smart contract-based context-aware access control scheme for blockchain-enabled IoT systems. *IEEE Internet Things J.* **11**(11), 19866–19881 (2024)
23. Sallam, K., Mohamed, M., Mohamed, A.W.: Internet of Things (IoT) in supply chain management: challenges, opportunities, and best practices. *Sustain. Mach. Intell. J.* **2**, 32 (2023)
24. Kouhizadeh, M., Saberi, S., Sarkis, J.: Blockchain technology and the sustainable supply chain: theoretically exploring adoption barriers. *Int. J. Prod. Econ.* **231**, 107831 (2021)

25. Iranmanesh, M., et al.: Effects of supply chain transparency, alignment, adaptability, and agility on blockchain adoption in supply chain among SMEs. *Comput. Ind. Eng.* **176**, 108931 (2023)
26. Azevedo, P., Gomes, J., Romão, M.: Supply chain traceability using blockchain. *Oper. Manag. Res.* **16**(3), 1359–1381 (2023)
27. Tian, F.: An agri-food supply chain traceability system for China based on RFID & blockchain technology. In: 2016 13th International Conference on Service Systems and Service Management (ICSSSM), Kunming, pp. 1–6. IEEE (2016)
28. Caro, M.P., Ali, M.S., Vecchio, M., Giaffreda, R.: Blockchain-based traceability in agri-food supply chain management: a practical implementation. In: 2018 IoT Vertical and Topical Summit on Agriculture - Tuscany (IOT Tuscany), Tuscany, Italy, pp. 1–4. IEEE (2018)
29. Abdallah, S., Nizamuddin, N.: Blockchain-based solution for pharma supply chain industry. *Comput. Ind. Eng.* **177**, 108997 (2023)
30. Liu, X., Barenji, A.V., Li, Z., Montreuil, B., Huang, G.Q.: Blockchain-based smart tracking and tracing platform for drug supply chain. *Comput. Ind. Eng.* **161**, 107669 (2021)
31. Reimers, T., Leber, F., Lechner, U.: Integration of blockchain and internet of things in a car supply chain. In: 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON), Newark, CA, USA, pp. 146–151. IEEE (2019)
32. Bruni, M.E., et al.: Decentralizing electric vehicle supply chains: value proposition and system design. In: 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1756–1761. IEEE (2023)
33. Al-Khatib, A.W.: Internet of things, big data analytics and operational performance: the mediating effect of supply chain visibility. *J. Manuf. Technol. Manag.* **34**(1), 1–24 (2023)
34. Francisco, F.E., Costa, A., Sampaio, P., Domingues, P., de Oliveira, O.J.: Implementation and improvement of integrated management systems: recommendations for their adaptation to the ISO high-level structure. *Cleaner Environ. Syst.* **15**, 100227 (2024)
35. International Organization for Standardization (ISO). <https://www.iso.org/home.html>. Accessed 4 Sept 2024
36. Haxhibeqiri, J., De Poorter, E., Moerman, I., Hoebeke, J.: A survey of LoRaWAN for IoT: from technology to application. *Sensors* **18**(11), 3995 (2018)
37. Almuhaya, M.A., Jabbar, W.A., Sulaiman, N., Abdulmalek, S.: A survey on Lorawan technology: recent trends, opportunities, simulation tools and future directions. *Electronics* **11**(1), 164 (2022)
38. Niles, K., Ray, J., Niles, K., Maxwell, A., Netchaev, A.: Monitoring for analytes through LoRA and LoRaWAN technology. *Procedia Comput. Sci.* **185**, 152–159 (2021)
39. Ethereum Foundation. What is Ether (ETH)? <https://ethereum.org/en/eth/>. Accessed 19 Aug 2024



Food Safety 4.0: The Future of Food Safety Leveraging Industry 4.0 Technologies

Eustace M. Dogo¹ , Tebogo Bokaba² , Hussaini A. Makun³ ,
Ahmed Aliyu¹ , and Ponjul Bali Kparbong¹

¹ Department of Computer Engineering, Federal University of Technology, Minna, Nigeria
eustace.dogo@futminna.edu.ng

² Department of Applied Information Systems, University of
Johannesburg, Johannesburg, South Africa

³ Africa Centre of Excellence for Mycotoxin and Food Safety, Federal University of
Technology, Minna, Nigeria

Abstract. The convergence of several factors, including population growth, scarce natural resources, climate change, globalization, sustainability, and advances in modern and emerging technologies, in addition to growing demand by consumers, retailers, regulators, and stakeholders to raise the level of food protection is increasingly driving the global food safety agenda. Industry 4.0 technologies are fast infusing into all fields of human endeavor including the food safety ecosystem, serving as catalysts for innovation and sustainability. However, there is no empirical evidence regarding the extent of their application and the level of acceptance within the food safety domain. This study explored the relationship between Industry 4.0 technologies and food safety by evaluating the applications of the Fourth Industrial Revolution technologies in addressing food safety challenges. The objectives are achieved using qualitative methodology and bibliometric analysis of content. Our analysis indicates that artificial intelligence, the Internet of Things, machine learning, and big data are prominent topics related to the 4IR and food safety, while blockchain and smart manufacturing are emerging topics.

Keywords: Food Safety · Industry 4.0 · Emerging Technologies · Sustainability · Systematic Review

1 Introduction

Food safety risks are evolving daily, such as the potential transfer of harmful microorganisms from diseased crops and/or animals, food additives, toxic substances, residues of veterinary drugs and other contaminants, and food fraud within the food value chain with consequences that negatively impact human health [1]. Industry 4.0 can offer various solutions to enhance food quality and safety, such as sensors, traceability, analytics, and automation.

Since research on 4IR technologies' applications in food safety is becoming prevalent, there arises a need to summarize the extant literature and reveal the domain's

intellectual structure. Furthermore, the fast-moving Fourth Industrial Revolution (4IR) high-tech technologies can greatly impact the food safety domain within a short period: hence, structured reviews need to periodically summarize the existing literature and identify important research gaps. To the best of our knowledge, to date, there are few reviews based on bibliometrics that discuss 4IR technologies' applications in the food safety domain. This presents a clear knowledge gap. Thus, it becomes important to make sense of the knowledge amassed over time to reveal new contributions, capture research trends, identify which topics are studied, and identify the potential research directions at the intersection of these two domains. Therefore, this study aimed to achieve the following research objectives: 1) Identify of the influential articles contributing to 4IR technologies in food safety, 2) Understand the evolution of linkages and citation networks over time, and 3) Cluster research themes and identify future research directions and emerging topics.

Motivated by the promising yet underexplored landscape of 4IR's role in food safety, this study undertook a comprehensive bibliometric review to map key research contributions, identify emerging trends, and propose directions for further advancement. By capturing the evolving technological landscape, this review aimed to consolidate existing knowledge and provide a structured foundation for future research initiatives in the agri-food sector.

The remainder of the paper is structured as follows: Sect. 2 examines global food safety regulations. Sections 3 and 4 briefly discuss the core 4IR technologies and their current applications in food safety. Section 5 discusses the technology acceptance models of the 4IR in the food safety domain. Section 6 outlines the methodology used in the bibliometric analysis. Section 7 presents and discusses the results obtained. Section 8 provides conclusions and future directions, while Sect. 9, the final section, outlines the limitations of the study.

2 Perspectives on Global Food Safety Regulations

North America's Food and Drug Administration (FDA) and the Canadian Food Inspection (CFIA) emphasize the significance of food safety laws for public health and global trade, supported by initiatives like the Regulatory Cooperation Council (RCC) and the United States-Mexico-Canada Agreement (USMCA) to harmonize regulations and ensure uniform standards [2]. Regulatory approaches align with international standards, focus on risk-based evaluations, and implement preventive controls. However, emerging food safety threats and regulatory disparities pose challenges, necessitating ongoing cooperation and innovation to enhance food safety systems.

Europe's stringent food safety regulations, overseen by the European Commission and the European Food Safety Authority (EFSA), ensure high standards across the food supply chain. European Union (EU) harmonization efforts and participation in international forums promote regulatory uniformity and transparency [3]. The precautionary principles guide risk management, while consumer engagement and traceability technologies boost transparency and empowerment. Despite facing new threats, Europe remains dedicated to innovation and continuous improvement in food safety regulations [4].

Asia, the Middle East and Africa (MEA) face diverse challenges in food safety due to varied regulatory frameworks, limited resources, and economic disparities. In Asia, efforts to comply with global standards are hindered by inadequate infrastructure and resources [5]. Similarly, the MEA region experiences risks from rapid economic growth, which necessitates strengthened regulatory frameworks and improved surveillance to address enforcement and contamination issues [6]. Meanwhile, South America faces challenges related to insufficient infrastructure, despite ongoing efforts to standardize regulations. To ensure food safety and protect public health across these regions, collaboration, adoption of international standards, and increased consumer awareness are essential [7].

3 The Fourth Industrial Revolution (4IR) Technologies

Figure 1 illustrates the core technologies of Industry 4.0, including augmented and virtual reality, Blockchain and nanoencapsulation, and artificial intelligence (AI), highlighting their interconnected roles in transforming modern industries.

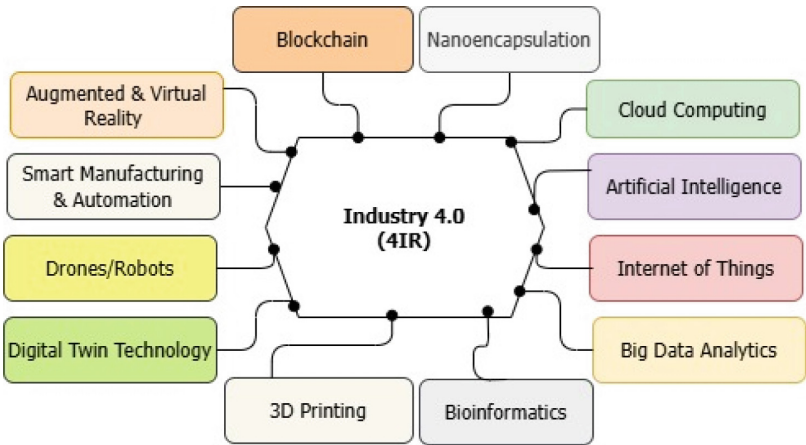


Fig. 1. A summarized coverage of the core 4IR technologies.

3.1 Emerging Digital Technologies in 4IR

The 4IR is powered by transformative technologies such as the Internet of Things (IoT), Industrial Internet of Things (IIoT), big data analytics (BDA), and AI. The IoT and IIoT connect devices and industrial systems to the Internet, generating data that optimizes processes, enhances urban planning, and improves industrial operations through real-time insights and predictive maintenance [8, 9].

BDA extracts actionable insights from vast datasets, intersecting with the IoT, AI, and cloud computing to drive operational efficiency and informed decision-making [10]. AI

further revolutionizes industries by enabling advanced automation and data-driven processes, fostering innovation, efficiency, and sustainability in manufacturing and beyond [11].

Cloud computing, a cornerstone of the 4IR, provides crucial infrastructure for data storage, processing, and access. Evolving variants such as edge and fog computing bring computation closer to data sources, enabling real-time decision-making and resource optimization in distributed environments [12]. These technologies drive innovations across various applications, from autonomous vehicles to smart cities. By facilitating seamless collaboration, data sharing, and remote work, cloud-based solutions present both opportunities and challenges, particularly in cybersecurity, requiring careful attention to privacy, data sovereignty, and compliance [13].

Complementing these advancements, virtual and augmented reality (VR and AR) are transforming industries and human experiences. VR immerses users in realistic training scenarios and remote collaboration, while AR enhances perception by overlaying digital information onto the real world, aiding in industrial maintenance, retail, education, and healthcare [14]. Extending these concepts, the Metaverse creates a shared 3D virtual space where users interact in real-time, integrating AI, big data, Blockchain, and AR/VR to revolutionize entertainment, work, and training [15]. Additionally, digital twin technology enhances the 4IR by creating virtual models of physical objects and systems, optimizing processes across industries from manufacturing to healthcare and smart cities [16].

Other notable technological advancements within the 4IR include Blockchain, nanoencapsulation technology, smart manufacturing, automation, preventative maintenance, robots, drones, 3D printing, new-generation communication technologies like 4G and 5G, and bioinformatics engineering. Blockchain ensures secure transaction tracking and data integrity, while nanoencapsulation technology enhances the stability and bioavailability of bioactive compounds in food [17, 18]. Smart manufacturing and automation utilize advanced technologies to improve efficiency, with preventative maintenance using IoT sensors for data-driven insights [19].

Additionally, robots and drones perform various tasks, enhancing efficiency in industries, while 3D printing allows for distributed manufacturing [20, 21]. New-generation communication technologies such as 4G and 5G support advancements in augmented reality, virtual reality, and the IoT despite challenges such as privacy concerns and equitable access [22]. Bioinformatics engineering merges biology and data analysis to revolutionize healthcare through personalized medicine, aiding in disease diagnosis and health risk prediction [23].

4 Current Applications of 4IR in Food Safety

The 4IR has significantly transformed food safety by leveraging data-driven insights, intelligent technologies, and collaborative initiatives. These advancements enhance quality standards, regulatory compliance, and consumer confidence across the entire food supply chain. Smart manufacturing utilizes Industry 4.0 technologies to improve production processes, reduce waste, and enhance resource efficiency, with smart sensors ensuring timely maintenance and minimal operational interruptions [19]. Supply chain

management benefits from Blockchain technology, which guarantees transparency and traceability, helping consumers and regulators authenticate the origin and safety of food items, thereby reducing fraud and improving operational effectiveness [24]. Additionally, BDA and the IIoT provide predictive insights and real-time monitoring of food processing and storage conditions, ensuring prompt corrective measures to prevent spoilage and bacterial proliferation [25].

Technologies such as radio frequency identification (RFID) tags, QR codes, and Blockchain enhance traceability and recall management, enabling precise identification of affected batches during food recalls, thus minimizing health risks [17]. VR and AR provide immersive training experiences for food safety practitioners, helping them simulate scenarios, practice hazard identification, and learn best practices [26]. Digital learning platforms further enrich practitioners' knowledge through accessible online courses, webinars, and e-learning materials, covering food safety regulations, risk assessment, and hygiene protocols [27].

Related literature reveals the transformative impact of 4IR technologies on food safety practices, significantly enhancing efficiency and safety across the agri-food sector. According to [28], intelligent biosensors, a core component of 4IR technology, are widely adopted for real-time monitoring and predictive analytics. Leveraging deep learning and the IoT, these biosensors elevate food safety risk management by facilitating traceability and empowering consumers to monitor compliance with safety standards. [29] explored advanced detection technologies, such as artificial nanochannel sensors, which further contribute by providing rapid and highly sensitive detection of biological and chemical threats in food, offering real-time results crucial for timely interventions. These technological benefits of 4IR innovations highlight a progressive shift in food safety practices.

In their studies, [30] and [31] highlighted that 4IR applications enhance supply chain management, streamlining operations to reduce waste while improving tracking and transparency in food safety protocols. Digital tools embedded within 4IR frameworks have fortified the resilience of food supply chains, especially in responding to disruptions like the COVID-19 pandemic, which underscored the importance of traceability and rapid response capabilities. The convergence of these technologies supports a robust framework that not only ensures higher safety standards but also prepares the sector to mitigate unforeseen risks. These reviews of related works showcase the significant strides made in integrating 4IR technologies within food safety.

5 Technology Acceptance Model of 4IR in Food Safety

The Technology Acceptance Model (TAM) and the Unified Theory of Acceptance and Use of Technology (UTAUT) are frameworks used for understanding technology adoption in food safety. The TAM examines perceptions of usefulness and ease of use, helping to identify barriers and facilitate the successful implementation of 4IR technologies in food safety [32]. TAM1, proposed by Davis in 1989, focuses on perceived usefulness and ease of use and has been applied to technologies like electronic monitoring systems in food storage [33]. TAM2, introduced by [34] in 2000, includes additional constructs such as subjective norm and job relevance, which are used to study the adoption of

traceability systems like RFID [35]. TAM3, proposed by [36] in 2008, further integrates constructs such as perceived enjoyment and social influence, aiding the understanding of emerging technologies such as AI-driven predictive analytics [37].

The UTAUT builds on the TAM, integrating various theories to offer a robust framework for understanding technology adoption. UTAUT1, proposed by [38], identifies performance expectancy, effort expectancy, social influence, and facilitating conditions, and has been applied to the adoption of digital food safety management systems [35]. UTAUT2, introduced in 2012, extends UTAUT1 by adding constructs like hedonic motivation, price value, and habit, making it relevant for understanding the acceptance of technologies such as mobile applications for real-time food safety monitoring [35]. These models help stakeholders comprehend factors influencing technology adoption, ultimately enhancing food safety practices.

6 Methodology

We conducted a bibliometric review using the PRISMA framework. Bibliometrics, a sub-branch of informatics, employs statistical techniques to measure the impact and dissemination of scientific publications. It allows for the analysis of large datasets with high accuracy, validity, transparency, and replicability [39]. This method identifies under-explored research areas and trends. The analysis focused on papers related to food safety and Industry 4.0 technologies, determine influential authors, topics, and publications and assess their impact. The employed methodology (the PRISMA process) is summarized in Fig. 2 and detailed in the subsequent subsections.

6.1 Search Sources

This study presents data collected from the Web of Science (WoS) and Scopus databases on June 1, 2024, covering the period 2015 to 2024. These databases were chosen for their extensive collection of abstracts and citations for academic articles, books, and conferences. They are considered representative of impactful research in this field. Additionally, both databases offer analytical tools to track, analyze, and visualize research data, and provide access to author profiles and publication metrics like the h-index and citation counts.

6.2 Data Search

The data was extracted using the following search terms: (*'Food Safety 4.0' AND 'Industry 4.0' OR '4IR' OR 'Artificial Intelligence' OR 'AI' OR 'Internet of Things' OR 'IoT'*

OR ‘Virtual Reality’ OR ‘Augmented Reality’ OR ‘VR’ OR ‘AR’ OR ‘3D Printing’ OR ‘Nanotechnology’ OR ‘Bio-technology’ OR ‘5G’ OR ‘Big Data’ OR ‘Cloud Computing’ OR ‘Digital Twin’ OR ‘Drones’ OR ‘Robots’). Using the search for the period (2015–2024), a total of 3,710 documents were retrieved from both databases.

6.3 Inclusion and Exclusion Criteria

As part of the screening and filtering process on the article search results, we conducted a thorough search of titles, keywords, and abstracts following the summarized criteria as outlined in Table 1.

Table 1. The inclusion and exclusion criteria for the selection process.

S/N	Inclusion criteria	Exclusion criteria
1	The Publication language is English	The articles are not written in English
2	The study utilizes empirical methodology focusing on reviews related to the 4IR within the domain of food safety	The article is a master’s dissertation, doctoral thesis, working paper preprint, or white paper
3	It includes concepts relating to the 4IR or associated smart technologies	The article does not specifically include concepts of the 4IR
5	The research is in an accredited journal or conference publication in subject areas such as Computer Science, Engineering, Decision Science, Mathematics, and Multidisciplinary	The authors’ country affiliations and their countries of study are not indicated and are not of the specified subject areas
6	The study is in a blind peer-reviewed publication	The articles do not use empirical research methods
7	The articles must be publications from 2015 to date	The articles are from publications prior to 2015

6.4 The Application of the PRISMA Guidelines

Following the PRISMA guidelines, a systematic search strategy was developed to identify relevant research papers from 2015–2024, focusing on terms such as “4IR,” “Industry 4.0”, and technologies such as “machine learning”, “Blockchain”, “IoT”, and “AR/VR” in food safety. As noted, the initial search retrieved 3,710 documents from the WoS and Scopus databases. These were filtered using the criteria outlined in Table 1, which included selecting English-language publications and excluding non-peer-reviewed publications. Titles, abstracts, and keywords were screened, narrowing the selection to 439 articles. A detailed full-text review further refined this to 232 articles. Figure 2 depicts the filtering process, highlighting the refinement from the initial pool to the final dataset.

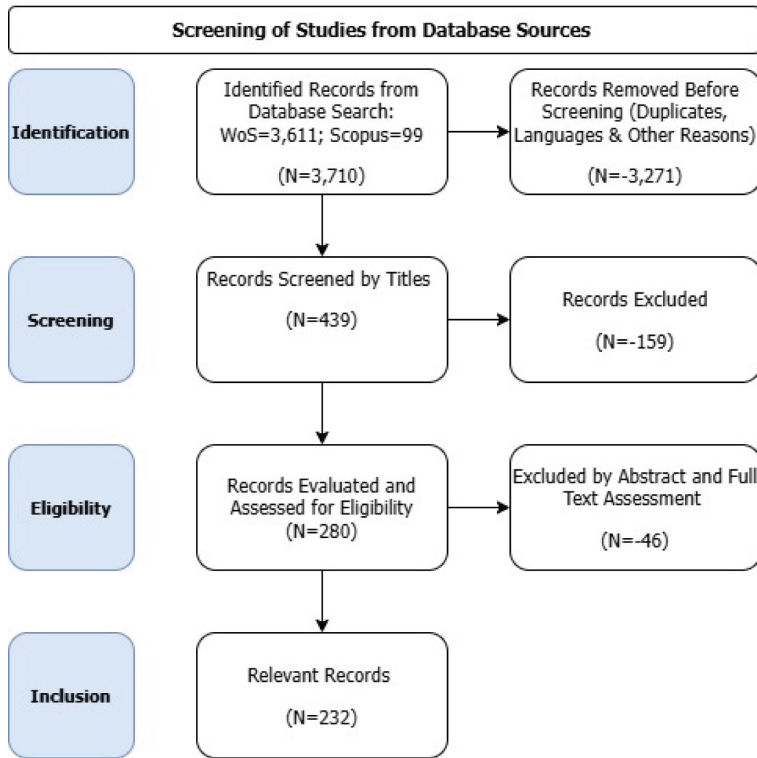


Fig. 2. The PRISMA process workflow.

6.5 Data Analysis

The data analysis was conducted using Rstudio (Biblioshiny) software, a versatile open-source tool for statistical computing and graphics. Renowned for its popularity in data analysis and statistical computing, Rstudio is widely used by researchers, data analysts, and practitioners for statistical analyses, visualizations, and generating data-driven insights. It integrates hundreds of packages and is frequently updated, making it ideal for meta-analyses like bibliometric studies [40]. Additionally, VOSviewer software was used for further analysis. VOSviewer, an open-source program, visualizes and analyzes research publication data, such as citation and co-authorship networks, and creates maps and visualizations of bibliometric data [41].

7 Result and Discussion

The search results are summarized in Table 2 and discussed in the subsequent subsections, namely, annual production by publication, most relevant sources and authors, productive affiliations, most productive countries, co-occurrence by authors' keywords, and analysis of relevant studies.

Table 2. Summary of the main information.

Description	WoS	Scopus
Main information about the data		
Timespan	2015:2024	2015:2024
Sources (Journals, Books, etc.)	579	67
Documents	3611	99
Annual growth rate %	61.36	27.1
Document average age	2.66	2.8
Average citations per doc	19.62	13.03
References	160962	3647
Document contents		
Keywords plus (ID)	3847	798
Author's keywords (DE)	9793	298
Authors		
Authors	10415	368
Authors of single-authored docs	219	8
Authors collaboration		
Single-authored docs	241	8
Co-Authors per document	3.76	3.96
International co-authorships %	31.29	25.25
Document types		
Article	2894	23
Book chapter	2	8
Book	123	2
Conference proceedings paper	56	48
Retracted publications	4	-
Correction	2	-
Editorial material	94	-
News item/Letter	3	1
Retracted publication	1	-
Review	440	9

7.1 Annual Production by Publication

Figure 3 shows the annual distribution of all 3,710 documents from 2015–2024. From 2015 to 2018, the production of articles indexed in both WoS and Scopus showed a significant upward trend. In 2015, there were only nine articles indexed in WoS and none in Scopus. By 2016, the numbers increased to 40 articles in the WoS and three

articles in Scopus and continued to increase to 80 articles and four articles in 2017, respectively. The most notable growth occurred in 2018, with 213 articles in WoS and six articles in Scopus. This data illustrates a strong positive correlation between the years and the number of articles published, reflecting a growing interest and research activity in the field.

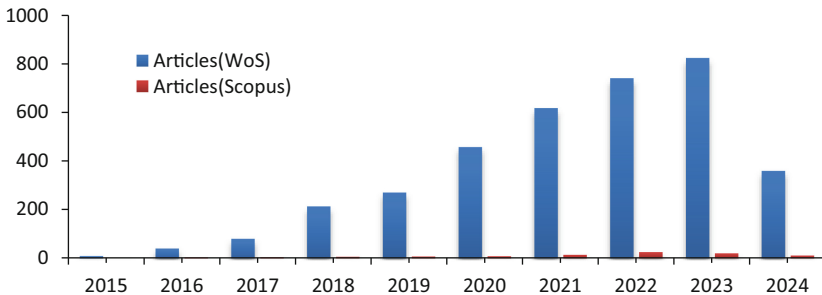


Fig. 3. Distribution of annual publication trend of the articles.

7.2 Most Productive Journals and Authors

Figures 4 and 5 provide an overview of key contributors and influential publications in 4IR and food safety. Figure 4 identifies the top 20 journals and conferences that are central to disseminating important research, helping researchers stay informed about high-quality studies and innovations in the sector. Figure 5 highlights the top 20 authors who have made significant impacts through their publications, showcasing their leadership and expertise in integrating 4IR technologies with food safety. This recognition underscores their role in advancing the field and promotes collaboration.

7.3 Most Productive Affiliations

Figures 6 and 7 illustrate the top institutions in 4IR technologies and food safety based on publications in the WoS and Scopus databases. In the WoS database, the Technical University of Kosice and the University of Minho lead with 100 articles each, followed by Politecn Milan, and Beijing Technology and Business University, which contribute fewer articles, while the University of Johannesburg and the University of Politecnica Valencia, have moderate outputs. The University of Naples Federico II, the University of Pisa, and the University of Calabria show the least contributions. In Scopus, the National Center for Toxicological Research is the leading contributor with nearly 10 articles, with other institutions showing varying levels of involvement.

7.4 Most Productive Countries

Table 3 shows that China leads in 4IR and food safety research with 9,315 citations and an average of 19.29 citations per article, emphasizing Asia's strong influence. Italy

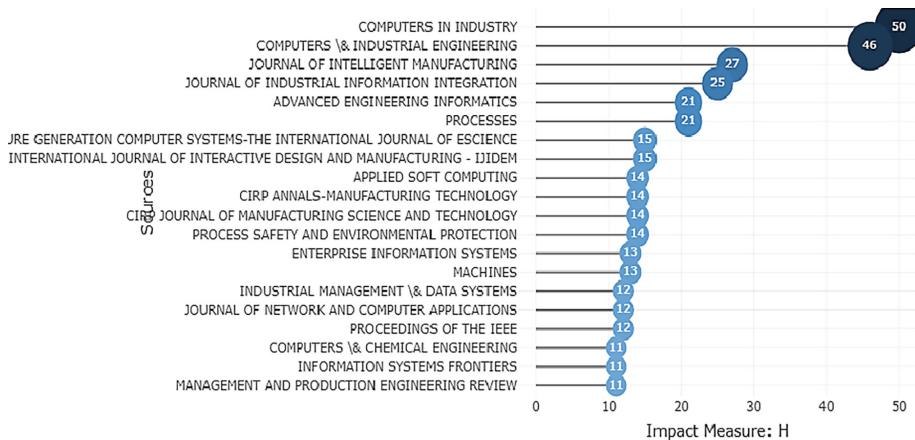


Fig. 4. Distribution of the key journals publishing in 4IR and food safety.

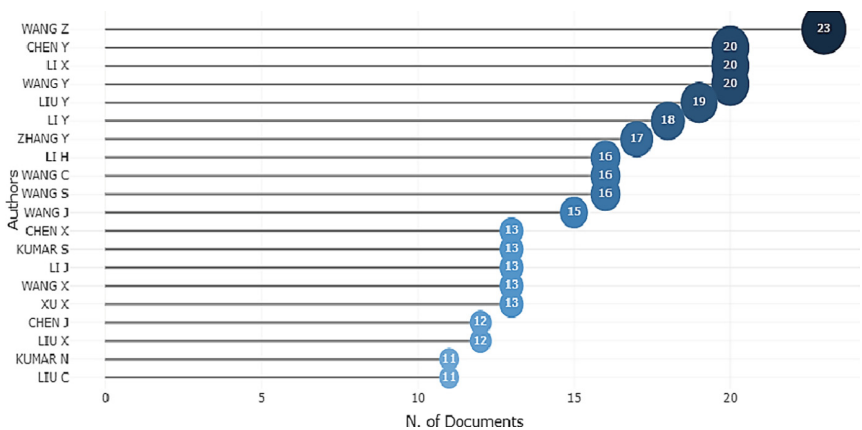


Fig. 5. Most productive authors in 4IR and food safety.

and Germany follow with 6,899 and 6,575 citations, respectively, with Germany having the highest average of 35.16 citations per article, indicating high-impact research. The United States of America (USA) has 5,357 citations and an average of 33.48 citations per article, reflecting significant contributions. Other notable contributors include the United Kingdom, India, Spain, and several other countries, highlighting a global research impact and collaborative efforts across continents in advancing 4IR technologies and food safety.

Table 4 highlights Europe’s dominance in research output with the highest total citations, reflecting its strong academic influence. Asia’s significant contributions, led by China and India, indicate a growing research presence. North America, particularly the USA, maintains a substantial research impact despite fewer total citations compared to Europe and Asia. South America, Africa, and Oceania, though having lower total citations, showcase diverse research interests and regional strengths. These figures can guide policy decisions and funding allocations, encouraging regions with lower citations

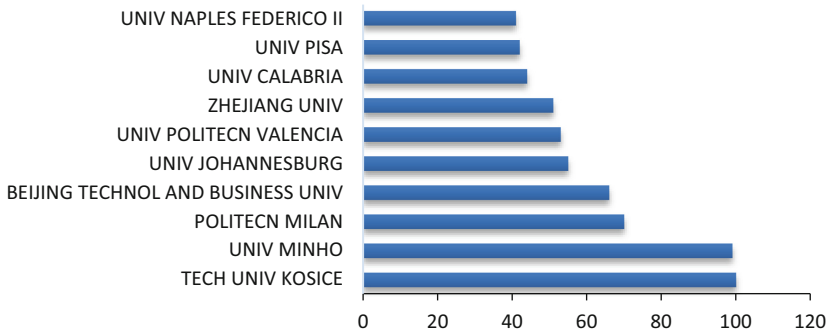


Fig. 6. Distribution of the top 10 most relevant affiliations from the WoS database.

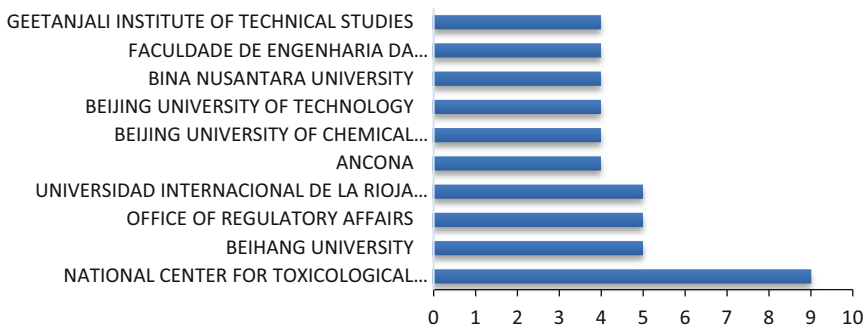


Fig. 7. Distribution of the top 10 most relevant affiliations from the Scopus database.

to invest more in research and development to enhance their academic impact in this research domain.

7.5 Co-occurrence by Author's Keywords

Most Frequent Authors' Keywords. The co-occurrence network analysis of author keywords, as shown in Fig. 8, reveals key trends in 4IR and food safety research. Prominent technologies include AI, the IoT, and machine learning, with AI showing a strong and increasing presence in recent studies. The IoT and machine learning also highlight significant research trends, particularly in predictive analytics and automation. Blockchain and big data are notable for enhancing transparency and decision-making, while AR and digital twin are emerging technologies gaining traction. The analysis also emphasizes the growing importance of sustainability and digital transformation in 4IR and food safety research, reflecting a shift towards environmentally friendly practices and the integration of advanced technologies in manufacturing.

Thematic Map. Figure 9 presents a thematic map of authors' keywords related to 4IR technologies and food safety, illustrating the research landscape. The map is divided into four quadrants: the lower-right quadrant shows basic themes such as machine learning,

Table 3. Top 20 most productive countries.

S/N	Country	Total citations	Average article citations	S/N	Country	Total citations	Average article citations
1	China	9,315	19.29	11	Brazil	2,120	19.45
2	Italy	6,899	22.77	12	Turkey	1,880	39.17
3	Germany	6,575	35.16	13	Korea	1,475	25.43
4	USA	5,357	33.48	14	Australia	1,457	25.56
5	UK	4,317	29.77	15	Switzerland	1,169	97.42
6	India	3,873	14.29	16	Poland	930	9.59
7	Spain	3,246	17.08	17	Malaysia	909	17.48
8	New Zealand	2,487	248.70	18	Singapore	877	58.47
9	Portugal	2,282	20.94	19	Greece	873	17.12
10	France	2,153	20.70	20	Canada	694	13.61

Table 4. Citations by continent.

S/N	Continent	Total citations
1	Asia	18,221
2	Europe	38,702
3	North America	6,051
4	South America	2,120
5	Africa	686
6	Oceania	3,944

Industry 4.0 and the IoT, which are foundational but not deeply developed; the upper-right quadrant features themes such as Industry 4.0, which are both well-developed and central to the field; the upper-left quadrant highlights niche themes such as deep learning, learning intelligence, AR, and safety reliability, which are specialized and cohesive but less broadly relevant; and the lower-left quadrant identifies emerging themes such as an early stage Industry 4.0, with low density and centrality, indicating areas with potential for future research growth. This thematic map highlights where research is focused and which areas are either emerging, foundational, or declining in importance.

7.6 Analysis of Examined Studies

An initial pool of 3,710 documents as retrieved, from which 232 highly relevant articles were selected for an in-depth analysis of 4IR technologies in food safety, as illustrated

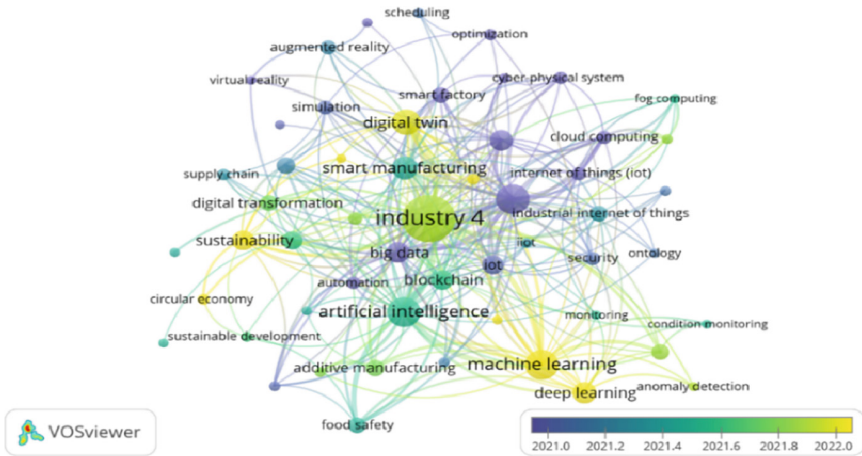


Fig. 8. The top 50 most occurring authors' keywords.

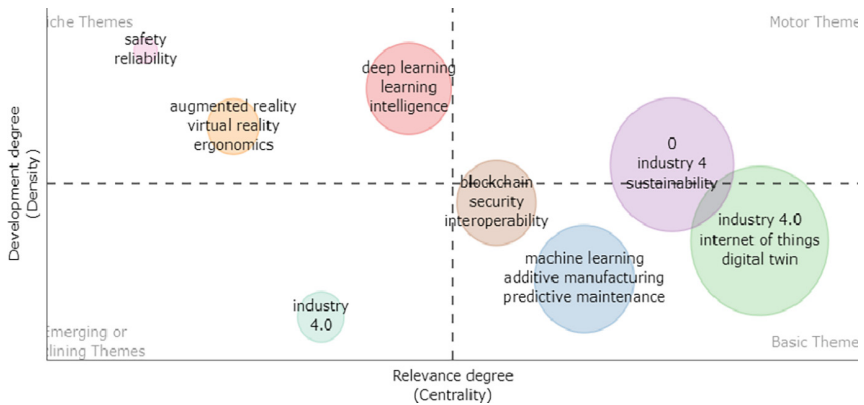


Fig. 9. Thematic map for keywords

earlier in Fig. 2. These selected documents were meticulously reviewed to extract and summarize critical insights on specific technologies, their applications, and key findings. A concise summary of the document types is presented in Table 5 below.

Table 5. The summarized highlight of information on the 232 articles.

Document type	Number of articles
Article	158
Review	53
Conference proceedings paper	14
Book chapter	4
Book	3

7.7 Annual Production

Figure 10 shows that from 2015 to 2024, the number of publications on 4IR technologies in food safety has generally increased. Beginning at a lower level in 2015 with 11 articles, publications rose steadily, peaking around 2018 with 24 articles. Although there was a brief decline after this peak probably due to the pandemic, the trend indicates a recovery and continued growth from 2022 (43 articles) onward, reflecting a rising interest and engagement in the field.

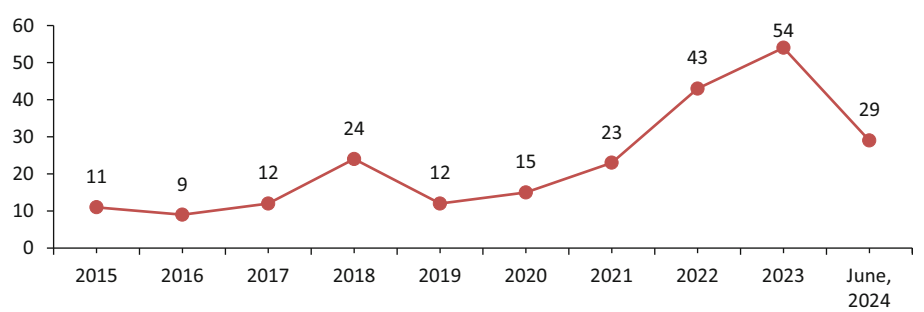


Fig. 10. Distribution of annual publication trend of the included article

7.8 Co-Occurrence of Authors' Keywords.

The distribution of keywords' frequency of occurrence shown in Fig. 11 emphasizes AI as a central node in research, closely linked with food safety, reflecting its significant role in these areas. AI's connections to Industry 4.0 and the IoT indicate its integration into industrial automation and smart technologies. The map also highlights emerging areas such as deep learning, Blockchain technology, precision agriculture, and robotics, where AI is driving advancements in data analytics, traceability, productivity, and sustainability. Additionally, terms such as sustainability and quality control are connected with AI, showing its application in promoting sustainable practices and enhancing product quality and safety.

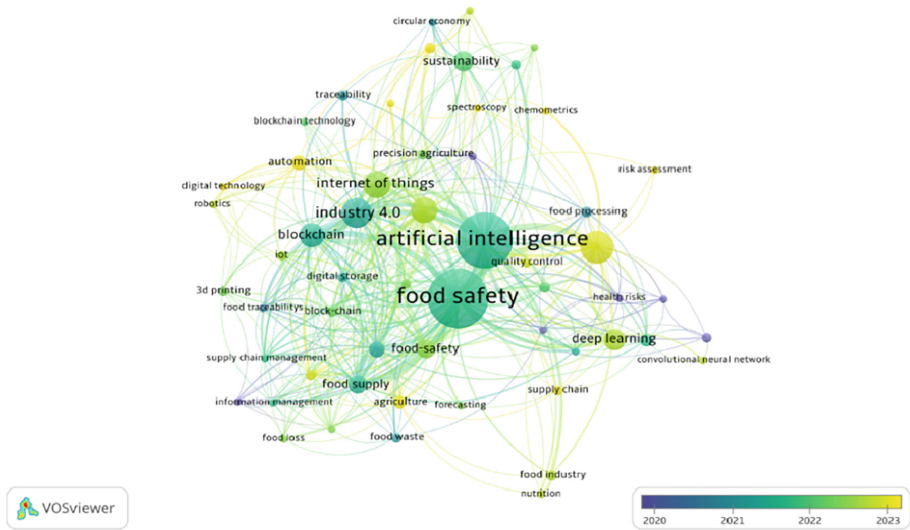


Fig. 11. Distribution of keywords occurrences.

Research classification on 4IR technologies that are enhancing food safety shows that AI is the most discussed technology, with 39 articles, confirming its significant role. Blockchain follows with 29 articles, and the IoT is covered in 12 articles, indicating its importance. BDA and bioinformatics are featured in eight and seven articles, respectively, showing their growing relevance. Automation and smart manufacturing appear in three articles, cloud computing, and AR and VR in two articles each, while digital twin and 3D printing technologies are each mentioned in one article, reflecting their emerging potential in food safety.

8 Conclusion and Future Direction

The research highlights key areas of focus within 4IR technologies for food safety, with AI (53%), the IoT (48%), and Blockchain (32%) dominating as popular fields from the 232 documents examined in this study. These technologies contribute significantly to predictive analytics, transparency, and traceability. BDA supports large-scale risk assessments, while digital twin technology (21%) shows potential for production monitoring and simulations. However, under-utilized areas such as nanoencapsulation (9%), robotics (15%), and VR and AR (6%) reveal opportunities for innovation. Bioinformatics and genomics, (less 3%) critical for pathogen detection, remain under-represented, suggesting these as promising directions for future research.

Despite their transformative potential, 4IR technologies in food safety face significant barriers. High operational costs and infrastructure demands associated with Blockchain and the IoT, particularly for smaller enterprises, challenge their adoption. Maintenance and computational power requirements further strain resources, while a shortage of skilled professionals complicates implementation. Addressing these barriers is essential

to fully realize their benefits, ensuring enhanced transparency, efficiency, and sustainability across the food supply chain. Overcoming these challenges will pave the way for accessible and impactful applications of 4IR in food safety, promoting broader adoption and long-term resilience.

9 Limitations of Study

A key limitation of this research study is the rapid evolution of 4IR technologies, which can make findings quickly outdated. As new advancements emerge, the relevance and applicability of current insights may diminish, limiting the study's long-term impact. To maintain accuracy and relevance, ongoing research is essential to capture these developments and update the understanding of 4IR's role in food safety. This dynamic nature highlights the need for periodic reviews to stay aligned with technological progress. Despite this limitation, the authors believe the findings of this study are a true representation of the current research trend in the application of 4IR technologies in the food safety domain.

References

1. Meyer, S.B., et al.: In the interest of food safety: a qualitative study investigating communication and trust between food regulators and food industry in the UK, Australia, and New Zealand. *BMC Public Health* **17**(1), 1–13 (2017)
2. Avitia-Carlos, P., Candolfi-Arballo, N., Luis Rodriguez-Verduzco, J., Rodriguez-Tapia, B.: Conditions for the Development and Certification of Industry 4.0 Technical Competencies. *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje* **17**(4), 336–342 (2022)
3. Bhat, R.: Emerging trends and sustainability challenges in the global agri-food sector. *Future Foods*, 1–21 (2022)
4. Qian, J., et al.: Food traceability system from governmental, corporate, and consumer perspectives in the European Union and China: A comparative review. *Trends Food Sci. Technol.* **99**, 402–412 (2020)
5. Sozinova, A.A., Sofiina, E.V., Petrenko, Y.S., Bencic, S.: International Features of Using Smart Technology in Agriculture: Overview of Innovative Trends. *Smart Innovation in agriculture*, 167–173 (2022)
6. Harris, K.J.: Global Food Safety: Perspectives, Challenges, and Plans. *Int. J. Food Sci. Nutr.* **3**(4), 1–5 (2016)
7. Bhat, Z.F., Morton, J.D., Mason, S.L., Bekhit, A.E.D.A., Bhat, H.F.: Technological, Regulatory, and Ethical Aspects of in Vitro Meat: A Future Slaughter-Free Harvest **18**(4), 1192–1208 (2019)
8. Kim, S.S., Kim, S.: Impact and prospect of the fourth industrial revolution in food safety: Mini-review. *Food Sci. Biotechnol.* **31**(4), 399–406 (2022)
9. Shaikh, M., Ali, A., Ahmed, R., Shaikh, B.A.: A Review on Internet of Things (IoT) Based Water Monitoring System. *Jurnal Kejuruteraan* **35**(6), 1273–1278 (2023)
10. Chakraborty, D., Rana, N.P., Khorana, S., Singu, B., Luthra, S.: Big Data in Food: Systematic Literature Review and Future Directions. *J. Comput. Inform. Syst.* **63**(5), 1243–1263 (2023)
11. He, C. et al.: A Review on Artificial Intelligence Enabled Design, Synthesis, and Process Optimization of Chemical Products for Industry 4.0. *Processes* **11**(2) (2023)

12. Dewangan, B.K., Agarwal, A., Choudhury, T., Pasricha, A., Chandra Satapathy, S.: Extensive review of cloud resource management techniques in industry 4.0: Issue and challenges. *Softw. Pract. Exp.* **51**(12), 2373–2392 (2021)
13. Jeyaraj, R., Balasubramaniam, A., et al.: Resource Management in Cloud and Cloud-influenced Technologies for Internet of Things Applications. *ACM Comp. Surv.* **55**(12), 1–37 (2023)
14. Liu, C., Xu, D., Dong, X., Huang, Q.: A review: Research progress of SERS-based sensors for agricultural applications. *Trends Food Sci. Technol.* **128**, 90–101 (2022)
15. Aromaa, S., Heikkilä, P., Kaasinen, E., Lammi, H., Tammela, A., Salminen, K.: Human factors and ergonomics considerations in the industrial metaverse. *Int. J. Human Fact. Ergon.* **11**(1), 4–27 (2024)
16. Pant, R., Singh, R., Gehlot, A., Akram, S. V., Gupta, L.R., Thakur, A.K.: A Systematic Review of Additive Manufacturing Solutions Using Machine Learning, Internet of Things, Big Data, Digital Twins and Blockchain Technologies: A Technological Perspective Towards Sustainability. *Archives of Computational Methods in Engineering*, 1–16 (2024)
17. Ellahi, R.M., Wood, L.C., Bekhit, A.E.D.A.: Blockchain-Based Frameworks for Food Traceability: A Systematic Review. *Foods* **12**(16), 3026 (2023)
18. Alazab, M., Alhyari, S.: Industry 4.0 Innovation: A Systematic Literature Review on the Role of Blockchain Technology in Creating Smart and Sustainable Manufacturing Facilities. *Inf.* **15**(2), 78 (2024)
19. Yan, W., Wang, J., Lu, S., Zhou, M., Peng, X.: A Review of Real-Time Fault Diagnosis Methods for Industrial Smart Manufacturing. *Processes* **11**(2), 369 (2023)
20. De Pace, F., Manuri, F., Sanna, A., Fornaro, C.: A systematic review of Augmented Reality interfaces for collaborative industrial robots. *Comput. Ind. Eng.* **149**, 106806 (2020)
21. Fasogbon, B.M., Adebo, O.A.: A bibliometric analysis of 3D food printing research: A global and African perspective. *Future Foods* **6**, 100175 (2022)
22. Santos, G.L., Endo, P.T., Sadok, D., Kelner, J.: When 5G Meets Deep Learning: A Systematic Review. *Algorithms* **13**(9), 208 (2020)
23. Ferrocino, I., et al.: The need for an integrated multi-OMICs approach in microbiome science in the food system. *Comprehensive Rev. Food Sci. Food Safety* **22**(2), 1082–1103 (2023)
24. Issa, A., Mekanna, A. N., Doumit, J., Bou-Mitri, C.: Redefining food safety: the confluence of Web 3.0 and AI technologies in the meat supply chain-a systematic review. *Int. J. Food Sci. Technol.* (2024)
25. Kumar, N., Banerjee, C., Bala, M.: Performance evaluation of blockchain food supply chain management system: JkBFMs. *Innov. Syst. Softw. Eng.* 1–6 (2024)
26. Belsito, C., Franciosi, C., & Di Pasquale, V.: The Impact of Augmented Reality Devices on Operator Performance in Manufacturing Contexts. *Manage. Prod. Eng. Rev.* **15**(1) (2024)
27. Abaee, M.: A systematic review of how cloud computing (CC) and the Internet of Things (IoT) can contribute towards achieving the sustainable development goals (SDGs), <https://ucw.arcabc.ca/islandora/object/ucw%3A321/datastream/PDF/view>, last accessed 2024/09/22
28. Chen, Y., Wang, Y., Zhang, Y., Wang, X., Zhang, C., Cheng, N.: Intelligent Biosensors Promise Smarter Solutions in Food Safety 4.0. *Foods* **13**(20), 235 (2024)
29. Xu, Y., et al.: Recent Advances of Food Hazard Detection Based on Artificial Nanochannel Sensors. *J. Agric. Food Chem.* **72**(21), 11900–11916 (2024)
30. Abideen, A.Z., Sundram, V.P.K., Pyeman, J., Othman, A.K., Sorooshian, S.: Food Supply Chain Transformation through Technology and Future Research Directions - A Systematic Review. *Logist.* **5**(4), 83 (2021)
31. Almelhem, M., Kurucz, A., Süle, E., Buics, L.: Agriculture 4.0 Applications in Supply Chain Management for Food Safety - A Structured Literature Review. (2022)

32. Ursavaş, Ö.F.: Technology Acceptance Model: History, Theory, and Application. In *Conducting Technology Acceptance Research in Education : Theory, Models, Implementation, and Analysis*, Springer International Publishing, pp. 57–91. Cham (2022)
33. Ma, Q., Liu, L.: The technology acceptance model: A meta-analysis of empirical findings. *J. Organiz. End User Comput. (JOEUC)* **16**(1), 59–72 (2004)
34. Venkatesh, V., Davis, F.: A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies. *Manage. Sci.* **46**, 186–204 (2000). <https://doi.org/10.1287/mnsc.46.2.186.11926>
35. Lai, P.C.: The literature review of technology adoption models and theories for the novelty technology. *J. Inf. Syst. Technol. Manag.* **14**(1), 21–38 (2017)
36. Venkatesh, V., Bala, H.: Technology Acceptance Model 3 and a Research Agenda on Interventions. *Decis. Sci.* **39**, 273–315 (2008). <https://doi.org/10.1111/j.1540-5915.2008.00192.x>
37. Saravanos, A., Zervoudakis, S., Zheng, D.: Extending the Technology Acceptance Model 3 to Incorporate the Phenomenon of Warm-Glow. *Inf.* **13**(9), 429 (2022)
38. Venkatesh, V., Morris, M., Davis, G., Davis, F.: User Acceptance of Information Technology: Toward a Unified View. *MIS Q.* **27**, 425–478 (2003). <https://doi.org/10.2307/30036540>
39. Bhuiyan, M.R.I., Akter, M.S.: Assessing the Potential Usages of Blockchain to Transform Smart Bangladesh: A PRISMA Based Systematic Review. *J. Inf. Syst. Inform.* **6**(1), 245–269 (2024)
40. RStudio: RStudio User Guide, <https://docs.posit.co/ide/user/>, last accessed 2024/08/07
41. VOSviewer: VOSviewer - Visualizing scientific landscapes. VOSviewer, <https://www.vosviewer.com/>, last accessed 2024/08/07



Soil Drainage Classification Using Machine Learning Models: A Comparative Study

Mouhamadou Lamine Diakhame¹(✉) , Diarra Bousso Dieng² ,
Joshua Nougaret³ , and Jonathan Mwaura³

¹ Gaston Berger University, Saint-Louis, Senegal
`diakhame.mouhamadou-lamine@ugb.edu.sn`

² Gaston Berger University, Saint-Louis, Senegal
`dieng.diarra-bousso@ugb.edu.sn`

³ Roux Institute At Northeastern University, Portland, ME, USA
`{nougaret.j, J.mwaura}@northeastern.edu`

Abstract. Soils play a critical role in sustaining ecosystems and ensuring food security. Effective soil water management, particularly through drainage, is crucial for both agricultural productivity and environmental sustainability. Given the importance of soil in food security, numerous studies have focused on soil classification. However, many rely on satellite imagery, making them inaccessible to most farmers. This study explores an alternative approach to soil classification using easily accessible soil properties, such as soil moisture, soil temperature, rainfall, and air temperature. The developed classification system categorizes soils into three drainage classes: Excessively Drained, Poorly Drained, and Well Drained. The analysis utilized a dataset from the Soil Climate Analysis Network (SCAN), comprising 15 years of environmental data from over 200 U.S. sites. Various machine learning models were employed for the classification task, including Logistic Regression, Random Forest, XGBoost, LightGBM, ANN, CatBoost, CNN, and SVM. Among these, CatBoost emerged as the best performer, achieving an accuracy of 89.77% in drainage classification, along with high precision, recall, and F1-score. These results provide valuable insights for enhancing agricultural decision-making and promoting sustainable land management, supporting data-driven agricultural and conservation policies.

Keywords: Soil Classification · Machine Learning · Agriculture · Soil Drainage

1 Introduction

Soils form the foundation of terrestrial ecosystems, playing a critical role in global food security, climate regulation, and biodiversity conservation. The health and

Supported by the United States Department of State (DoS): Embassy in Dakar, under the University Partnership Initiative, Grant No. SSG20023GR0032.

functionality of soils are inherently tied to their ability to manage water, a characteristic primarily determined by drainage properties. With increasing pressure from human activities on natural resources, it is essential to understand and accurately classify soil drainage [18] to ensure sustainable land management and effective agricultural practices.

A major challenge in modern agriculture is soil degradation, exacerbated by intensive farming, urbanization, and climate change. This degradation undermines the soil's ability to retain nutrients, store water, and support diverse plant life. Such degradation not only threatens food production but also contributes to environmental problems, including increased greenhouse gas emissions and reduced biodiversity. In this context, accurately classifying soil drainage is essential for optimizing crop selection and rotation strategies, designing efficient irrigation systems, implementing targeted nutrient management plans, mitigating soil erosion and runoff, enhancing carbon sequestration potential, and preserving and restoring ecosystem services.

The work reported in this paper aims to address the development of a soil classification method. Unlike previous studies that focus predominantly on parameters such as nutrient levels (such as nitrogen (N), phosphorus (P), and potassium (K)), temperature, humidity, and pH for crop recommendations, this research emphasizes intrinsic soil properties, specifically soil type and drainage. These characteristics are more reflective of real field conditions, providing a more precise foundation for crop recommendations tailored to specific environmental contexts. By incorporating soil drainage and type, the study enables better-informed decisions, enhancing the effectiveness of agricultural practices.

The application of machine learning models in this context offers several benefits: it facilitates the analysis of large datasets and provides a scalable, cost-effective alternative to traditional soil classification methods. This approach presents new possibilities for democratizing access to crucial soil information, particularly for farmers and land managers in regions with limited resources for comprehensive soil surveys.

The reported work compares the performance of various machine learning models for soil drainage classification. These models include Logistic Regression, Random Forest, XGBoost, LightGBM, Artificial Neural Networks (ANN), CatBoost, Convolutional Neural Networks (CNN), and Support Vector Machine (SVM). The objective is to identify the most effective techniques among these methods for accurate soil drainage classification.

The significance of this research extends beyond academic interest: accurate large-scale soil drainage classification can improve agricultural productivity and sustainability, support better water resource management, and contribute to evidence-based policymaking for land use and conservation.

The rest of this article is formulated as follows: Sect. 2 presents a review of available literature in soil classification. In Sect. 3, the methodology followed in this work is presented, including the utilized dataset and the machine learning models. Section 4 presents the obtained results, while Sect. 5 gives a discussion

on the implication of the results. Finally, Sect. 6 draws a conclusion and presents our thoughts on the future direction of the work.

2 Related Work

Inazumi et al. [1] presented their research on the application of deep learning techniques [14] to classify soil types such as clay, sand, and gravel. Their aim was to mitigate the subjective nature of traditional soil classification [15] methods that often depend on the expertise of engineers. To achieve this, the authors developed a convolutional neural network (CNN) [16] trained on soil images taken with a smartphone camera. This model processes these images to create a practical AI-based solution for on-site soil classification. The CNN achieved a training accuracy of 86% and a verification accuracy of 77%, with sand classified most accurately (recall of 1.0), while gravel presented challenges (recall of 0.54). The study underscores issues such as imbalanced datasets and emphasizes the necessity for larger image sizes to enhance model generalization. In comparison to this work, the reported work does not utilize images; instead, the work focuses on features that do not require image processing.

Similarly, Mohd Azmin et al. [2] explored the use of machine learning algorithms [17] for soil classification aimed at optimizing crop recommendations. The study sought to overcome the limitations of existing soil classification techniques that heavily rely on expert knowledge and are time-consuming. The authors implemented three machine learning algorithms: Random Forest, Naïve Bayes, and k-Nearest Neighbors (k-NN). The dataset used was compiled from sources such as Kaggle and iGROW, containing various soil attributes like nitrogen, phosphorus, potassium, temperature, humidity, and pH levels. Results revealed that Random Forest outperformed the other models, achieving an accuracy of 97.23%, followed closely by Naïve Bayes at 96.82% and k-NN at 92.92%. Additionally, Random Forest had the highest kappa value of 0.971, indicating its superior accuracy and reliability in crop suggestion applications. The effectiveness of machine learning in soil classification and crop suggestion is emphasized, with Random Forest emerging as the most dependable model. With regard to this work, the reported work also compares Random Forest with other machine learning models.

In another related study, Beucher et al. [3] examined the utilization of Artificial Neural Networks (ANNs) and Decision Tree Classification (DTC) methods to predict soil drainage classes in Denmark. Their research mapped five drainage classes—very well-drained (DC1), well-drained (DC2), moderately well-drained (DC3), poorly drained (DC4), and very poorly drained (DC5)—which are vital for agricultural and environmental management. Utilizing a dataset of 1702 soil observations across Denmark, the study incorporated 31 environmental variables, including soil attributes and terrain characteristics, to enhance the models' predictive capabilities. The performance comparison showed that ANNs had a slight edge over DTC models, with accuracies of 54% vs. 52%, respectively. Notably, DTC models exhibited greater resilience to missing or redundant data,

whereas ANNs benefited from effective variable selection. Despite the higher computational costs associated with ANNs, both methodologies yielded comparable drainage maps, showcasing their potential for large-scale soil classification.

The article “Farmers Knowledge on Indigenous Soil Types: A Review” [4] explores how farmers use traditional knowledge to classify and manage soils. This indigenous knowledge, passed down through generations, helps farmers identify soil types based on visual and physical characteristics such as color, texture, and water retention. The article highlights the importance of this local knowledge for sustainable agriculture, as farmers often have a deep understanding of their environment. It also emphasizes the need to integrate indigenous soil classification systems with scientific approaches to enhance soil management practices. This integration can lead to improved productivity and environmental sustainability, especially in regions where scientific soil assessments may not be readily available. The review concludes that acknowledging farmers’ indigenous knowledge is crucial for formulating effective agricultural policies and that collaborative efforts between local farmers and researchers can enhance soil health and agricultural resilience.

In contrast to previous studies, this research specifically focuses on the classification of soils into three distinct drainage categories: Excessively Drained, Poorly Drained, and Well Drained. A comprehensive dataset sourced from the Soil Climate Analysis Network (SCAN) was utilized, comprising 15 years of daily measurements from over 200 sites across diverse geographical regions in the United States. This dataset includes critical environmental variables such as air temperature, rainfall, and soil moisture at various depths, providing a robust foundation for the classification task. Several advanced machine learning models were evaluated and compared, including Logistic Regression, Random Forest, XGBoost, LightGBM, Artificial Neural Networks (ANN), CatBoost, Convolutional Neural Networks (CNN), and Support Vector Machine (SVM). This work not only contributes to enhancing agricultural decision-making but also addresses limitations found in previous research by leveraging extensive and representative environmental data.

3 Methodology

3.1 Dataset

The dataset used in this study was obtained from the Soil Climate Analysis Network (SCAN) [13], a nationwide network that monitors soil moisture and climate conditions across the United States. SCAN provides 15 years of daily measurements from over 200 sites, encompassing a wide range of geographical regions, soil types, and climates. The dataset includes key environmental variables such as air temperature (in degrees Fahrenheit), rainfall (measured in inches), soil temperature (measured at 2 in, 4 in, 8 in, 20 in, and 40 in depths), and soil moisture, measured at the same depths as the soil temperature. The mentioned features are crucial for understanding the interactions between soil properties and drainage behavior.

To prepare the data for use with the machine learning models, preprocessing techniques were applied to manage missing values, remove outliers, and normalize the data, with a specific focus on capturing the soil's response during significant rainfall events. This preprocessing enhances the predictive performance of the machine learning models used for classifying soil drainage categories.

3.2 Machine Learning Models

In this research, a variety of machine learning models were implemented to classify soils into three drainage categories: Excessively Drained, Poorly Drained, and Well Drained. Each model was chosen for its ability to handle the complexities and high dimensionality inherent in the environmental data, as well as its potential to improve predictive accuracy. The following section details the models used and explains their application in the study.

Random Forest: Random Forest is a strong ensemble method that creates several decision trees and combines their predictions. This method proved particularly effective in processing the high-dimensional SCAN dataset, which includes numerous environmental variables such as soil moisture, temperature, and rainfall. By averaging the outputs of various decision trees, Random Forest mitigates overfitting and provides reliable predictions for soil drainage classification [5]. In the reported work, **Random Forest** classifier was configured with 100 decision trees (`n_estimators`) and a maximum depth of 10, ensuring a balance between model complexity and performance, with a fixed `random_state` of 42.

XGBoost: Extreme Gradient Boosting (XGBoost) is a highly effective algorithm used for gradient boosting. XGBoost was employed to handle the large SCAN dataset, utilizing its ability to manage missing values and built-in regularization techniques. The iterative improvement process of XGBoost allowed for accurate modeling of the complex interactions between soil properties and drainage outcomes [6]. The reported work uses the `multi:softmax` objective to handle multi-class classification with 100 boosting rounds (`n_estimators`) and a learning rate of 0.1. The maximum tree depth was set to 6, and the model optimized its predictions using the `mlogloss` evaluation metric.

LightGBM: LightGBM, another gradient boosting framework, was selected for its computational efficiency and ability to handle large datasets. Using a histogram-based approach, LightGBM accelerated the training process and minimized memory usage. This made it particularly useful for soil drainage classification, where computational efficiency is critical, and allowed for efficient processing of the diverse environmental features in the SCAN dataset [7]. The LightGBM model was set up for multi-class classification, using a learning rate of 0.1 and 100 boosting rounds. The maximum depth of the trees was set to 10, and the `random_state` was fixed at 42 to ensure repeatability.

Logistic Regression: Logistic Regression is a statistical method applied to binary classification, modeling the relationship between a binary outcome variable and one or more predictor variables. In this study, Logistic Regression was applied as

a baseline model for comparison against more complex techniques. Its simplicity and interpretability made it useful for understanding the impact of individual soil properties on drainage classification [8]. The **Logistic Regression** model was configured with the **multinomial** option for multi-class classification, using the **lbfgs** solver to optimize the likelihood function. Class imbalances were addressed using custom class weights, and the model was set to run for a maximum of 500 iterations with a random seed of 42 for reproducibility.

Artificial Neural Networks (ANN): Artificial Neural Networks (ANNs) were employed to capture non-linear relationships between environmental variables and soil drainage categories. ANNs are well-suited for detecting complex patterns in data, and in this study, they helped reveal intricate relationships between soil moisture at different depths, rainfall events, and drainage behavior. The adaptability of ANNs allowed for fine-tuning the model architecture to improve classification accuracy [9]. In the reported work, the ANN had two hidden layers, each containing 64 units with the **ReLU** activation function, followed by a softmax output layer for multi-class classification. The model was optimized using the **Adam** optimizer and trained with categorical cross-entropy as the loss function.

CatBoost: CatBoost is a gradient boosting algorithm specifically designed to handle categorical data, a common feature in environmental datasets. In this study, CatBoost reduced the need for extensive preprocessing by automatically managing categorical features in the SCAN dataset, improving the model's performance in predicting soil drainage categories [10]. The CatBoost model was trained with 1000 iterations and a learning rate of 0.1. The depth of the decision trees was set to 10, and the **random_state** was fixed at 42 for consistency in results.

Convolutional Neural Networks (CNN): Convolutional Neural Networks (CNNs) were adapted for this study to capture spatial and temporal relationships within the dataset. While CNNs are traditionally used for image analysis, they were employed here for structured data. By analyzing soil moisture and temperature at different depths, CNNs detected spatial patterns that contributed to more accurate soil drainage classification [11]. In the reported work, the CNN architecture included two **Conv1D** layers with 64 and 32 filters, followed by **MaxPooling** layers and one dense hidden layer with 64 units. The output layer employed a softmax activation function for classification.

Support Vector Machine (SVM): Support Vector Machines (SVMs) with the RBF kernel were used to classify soil drainage. Despite using scaled features like soil moisture and temperature, the model achieved a modest 63.04% accuracy, highlighting the need for better feature engineering and kernel selection in future studies.

Table 1 provides a summary of the key hyperparameters used for the machine learning models in this study.

In summary, the selected hyperparameters were aimed at achieving the best trade-off between model performance and computational efficiency. Each hyperparameter choice was informed by the specific characteristics of the dataset and the strengths of the models used. For instance, in models like Random Forest and

Table 1. Summary of Hyperparameters and Training Times for Machine Learning Models

Model	Hyperparameter	Value	Training Time (s)
Logistic Regression	multi_class	multinomial	1
	solver	lbfgs	
	class_weight	Custom (to handle class imbalance)	
	max_iter	500	
XGBoost	random_state	42	12
	objective	multi:softmax	
CatBoost	num_class	Number of classes (from label encoder)	666
	eval_metric	mlogloss	
	n_estimators	100	
	learning_rate	0.1	
	max_depth	6	
	random_state	42	
	iterations	1000	
ANN	learning_rate	0.1	493
	depth	10	
	random_state	42	
	verbose	False	
	Hidden Layers	2 hidden layers (64 units each, ReLU activation)	
CNN	Output Layer	Softmax (for multi-class classification)	1184
	Optimizer	adam	
	Loss	categorical_crossentropy	
	Metrics	accuracy	
	Conv1D Layers	2 Conv1D layers (64 and 32 filters)	
Random Forest	Pooling Layers	MaxPooling1D (pool size = 2)	87
	Dense Layers	1 hidden layer (64 units, ReLU activation)	
	Output Layer	Softmax (for multi-class classification)	
	n_estimators	100	
LightGBM	max_depth	10	11
	random_state	42	
	objective	multiclass	
SVM	num_class	Number of classes (from label encoder)	2022
	learning_rate	0.1	
	n_estimators	100	
	max_depth	10	
	random_state	42	
	kernel	rbf	
	random_state	42	
	C	Default	
	gamma	Default	

XGBoost, the number of trees and depth were chosen to avoid overfitting while maintaining predictive accuracy. In contrast, for LightGBM, the histogram-based approach made it ideal for large datasets, where the learning rate and boosting rounds were set to ensure fast convergence without sacrificing performance.

For ANNs and CNNs, the choice of hidden layers and units, as well as activation functions, was aimed at capturing the non-linear relationships in the data while ensuring sufficient model complexity to avoid underfitting. Logistic Regression was used as a baseline, with hyperparameters like class weight adjustments to handle class imbalances effectively. CatBoost was particularly chosen for its ability to handle categorical data efficiently, with hyperparameters tuned to optimize its performance for the specific soil drainage classification task.

Overall, the parameters were selected based on prior knowledge of the models' characteristics, ensuring that each model could best leverage the dataset's high dimensionality and complex relationships while maintaining computational feasibility and model interpretability.

3.3 Training of Machine Learning Models

The dataset was divided into two parts: 80% allocated for training and 20% reserved for testing. This split was chosen as a standard practice in machine learning to balance the need for a sufficient amount of data for model training with the requirement to reserve enough data for evaluating the model's generalization performance. An 80/20 split ensures that the training set is large enough for the model to learn effectively while the test set remains sufficiently representative of the data's variability, capturing potential edge cases.

Table 2 shows how the samples are distributed among the drainage classes in the training set. The "Well" drainage class has the highest number of samples (50,288), followed by the "Poorly" drained class (10,800 samples) and the "Excessively" drained class (7,600 samples). This distribution reveals a noticeable class imbalance, with the "Excessively" drained class being significantly underrepresented compared to the other classes.

While no specific measures were taken to address this imbalance during the training phase, its potential impact on model performance is acknowledged. Models trained on imbalanced datasets can exhibit a bias towards the majority class, potentially leading to lower precision and recall for the underrepresented "Excessively" drained class. In future iterations of this work, strategies to address this issue will be explored. These may include resampling techniques such as oversampling the minority class or undersampling the majority class and using algorithms designed to handle class imbalances effectively.

Despite these challenges, the chosen 80/20 split is believed to provide a reasonable trade-off between model training and testing, allowing for an adequate evaluation of the classifier's performance across all classes. The metrics presented in this study provide insight into how well the model handles the imbalance, highlighting areas for potential improvement.

Table 2. Number of Samples per Class in the Training Set

Drainage Class	Well	Poorly	Excessively
Number of Samples (Training)	50,288	10,800	7,600

Table 3 presents the distribution of samples in the test set, where the “Well” class again has the highest number of samples at 12,573, followed by ‘Poorly’ with 2,700, and “Excessively” with 1,900 samples.

Table 3. Number of Samples per Class in the Test Set

Drainage Class	Well	Poorly	Excessively
Number of Samples (Test)	12,573	2,700	1,900

3.4 Evaluation of Models

To ensure the robustness of the reported results, each model was trained and tested using the dataset split into training and testing sets. The models were evaluated using standard classification metrics, including accuracy, precision, recall, and F1-score [12].

4 Results

This section presents the results obtained by applying various machine learning techniques to the provided dataset for classifying soils into excessively drained, poorly drained, and well-drained.

Table 4 presents the precision scores for each model across different soil classes. CatBoost achieved the highest precision for the ”Excessively” class at **0.86**, indicating it correctly identified most of the instances in this class. In contrast, the Logistic Regression model struggled significantly, with a precision of only **0.25** for the same class. The macro and weighted averages show that CatBoost not only excels in precision but also maintains overall performance across the board.

Table 5 displays the recall metrics, which reflect the models’ ability to correctly identify instances of each soil class. CatBoost again shows strong performance with a recall of **0.76** for the “Excessively” class. Notably, Random Forest excels in the “Poorly” class with a recall of **0.86**, indicating its effectiveness in recognizing this particular category. The macro average recall values indicate that most models perform comparably well across classes, with CatBoost and LightGBM leading.

Table 4. Precision for Different Models

Class	CatBoost	LightGBM	XGBoost	Random Forest	Logistic Regression	CNN	ANN	SVM
Excessively	0.86	0.71	0.63	0.51	0.25	0.55	0.58	0.72
Poorly	0.81	0.67	0.60	0.48	0.40	0.64	0.59	0.75
Well	0.92	0.91	0.91	0.92	0.85	0.91	0.93	0.79
Macro Avg	0.87	0.76	0.71	0.64	0.50	0.70	0.70	0.80
Weighted Avg	0.90	0.85	0.83	0.81	0.71	0.83	0.84	0.87

Table 5. Recall for Different Models

Class	CatBoost	LightGBM	XGBoost	Random Forest	Logistic Regression	CNN	ANN	SVM
Excessively	0.76	0.72	0.76	0.80	0.75	0.82	0.81	0.22
Poorly	0.78	0.77	0.78	0.86	0.79	0.74	0.84	0.37
Well	0.94	0.87	0.82	0.68	0.42	0.81	0.79	0.96
Macro Avg	0.83	0.79	0.79	0.78	0.65	0.79	0.81	0.81
Weighted Avg	0.90	0.84	0.81	0.75	0.52	0.80	0.80	0.82

Finally, Table 6 provides the F1-Score for each model, which balances precision and recall into a single metric. CatBoost achieves the highest F1-Score of **0.81** for the “Excessively” class, reflecting its overall effectiveness in classification. The F1-Scores indicate that while all models generally perform well, CatBoost consistently maintains the best balance between precision and recall across the soil classes. This demonstrates its robustness in handling various classification tasks.

Table 6. F1-Score for Different Models

Class	CatBoost	LightGBM	XGBoost	Random Forest	Logistic Regression	CNN	ANN	SVM
Excessively	0.81	0.71	0.69	0.63	0.38	0.66	0.68	0.34
Poorly	0.80	0.72	0.68	0.61	0.53	0.68	0.69	0.50
Well	0.93	0.89	0.86	0.79	0.57	0.86	0.85	0.87
Macro Avg	0.85	0.77	0.74	0.67	0.49	0.73	0.74	0.78
Weighted Avg	0.90	0.84	0.81	0.74	0.54	0.81	0.81	0.82

In Table 7, the accuracy scores of all models are compared to provide a clear overview of their performance. This comparison of machine learning models allows us to identify the most reliable approach for soil drainage classification. Our findings suggest that CatBoost, with its superior handling of complex environmental data, is the most suitable model for this application.

Figure 1 shows the confusion matrix of the CatBoost model, illustrating its performance in classifying different soil types. CatBoost achieved the highest performance with an accuracy of 89.77%, along with precision, recall, and F1-score metrics, demonstrating its effectiveness in classifying soil drainage categories.

Table 7. Model Accuracy Comparison

Model	Accuracy
CatBoost	89.77%
LightGBM	84.03%
XGBoost	80.82%
Random Forest	72.45%
Logistic Regression	51.66%
CNN	79.93%
ANN	79.96%
SVM	63.04%

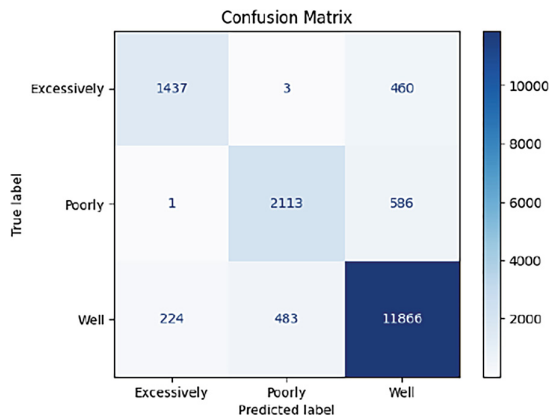


Fig. 1. Confusion matrix of the CatBoost model.

5 Discussion

This study evaluates the performance of several machine learning models in classifying soil types into drainage categories: Excessively Drained, Poorly Drained, and Well Drained. The results reveal notable differences in accuracy across the models, providing insights into their effectiveness in soil classification tasks.

CatBoost emerged as the top-performing model, achieving an accuracy of **89.77%**, significantly outperforming the other models. Its gradient boosting framework, which combines multiple weak learners, enables it to capture complex patterns in the data, making it particularly effective for this task.

LightGBM followed with an accuracy of **84.03%**, showing strong performance but slightly trailing CatBoost. LightGBM’s ability to handle large datasets efficiently and its use of leaf-wise tree growth help explain its high accuracy, although it may be more sensitive to hyperparameter tuning, which can explain the difference in performance compared to CatBoost.

XGBoost achieved an accuracy of **80.82%**, placing it third in terms of model performance. While it shares similarities with CatBoost and LightGBM in terms of its boosting approach, XGBoost's lower accuracy suggests that its hyperparameters may require further tuning for optimal performance in this specific task.

Support Vector Machine (SVM) demonstrated an accuracy of **63.04%**, which placed it among the lower performers. Despite being a powerful classifier for binary classification tasks, SVM struggles with multi-class problems like soil drainage classification. The confusion matrix reveals that SVM performed relatively well in classifying "Well Drained" soils, but its performance was weaker in distinguishing between "Excessively Drained" and "Poorly Drained" categories. This is evident from the precision and recall values, with higher recall for "Well Drained" and lower recall for "Excessively Drained" and "Poorly Drained".

Random Forest displayed a moderate accuracy of **72.45%**, demonstrating its reliability in handling this classification problem. However, compared to gradient boosting models, Random Forest struggled to capture the more intricate relationships in the dataset, leading to lower overall performance.

Logistic Regression performed poorly in comparison, achieving an accuracy of only **51.66%**. This reflects the model's limitations in handling non-linear relationships in the data, as logistic regression is better suited to linearly separable problems, which is not the case for soil drainage classification.

Convolutional Neural Networks (CNNs) and **Artificial Neural Networks (ANNs)** showed similar performance, with accuracies of **79.93%** and **79.96%**, respectively. While these models are generally effective at learning complex patterns, their performance lagged behind CatBoost and LightGBM. This may be due to the specific architecture used or the need for more extensive hyperparameter tuning and training. Their competitive results suggest that neural networks remain viable options, though they may require more tailored configurations for this problem.

In summary, while **CatBoost** emerged as the most accurate model in this study, **LightGBM** and **XGBoost** also performed well, highlighting the overall effectiveness of gradient boosting techniques for soil drainage classification. Traditional models like **Random Forest** and **Logistic Regression** showed limitations, particularly when compared to more advanced methods. Neural networks (**CNN** and **ANN**) performed moderately well but did not surpass the boosting models, suggesting room for further refinement. **SVM**, though reliable for certain types of classification tasks, showed lower performance due to its struggle with multi-class classification in this context. Future research should explore the potential of ensemble methods or hybrid models to improve classification accuracy and robustness in soil science applications.

6 Conclusion

This study demonstrates the transformative impact of machine learning on soil drainage classification using environmental data. Using a comprehensive dataset

from the Soil Climate Analysis Network (SCAN), several models were evaluated, with CatBoost emerging as the best performer, achieving an accuracy of 89.77% in drainage classification, along with high precision, recall, and F1-score. Its ability to handle categorical data efficiently and uncover complex patterns underscores its superiority. LightGBM and XGBoost also delivered strong results, reinforcing the reliability of gradient boosting methods for this application.

The findings carry significant implications. Improved soil drainage classification enhances agricultural decision-making by optimizing crop selection, irrigation, and nutrient management, thus improving productivity and resource use efficiency. From an environmental perspective, these methods enable better land use planning and soil conservation, supporting climate resilience by informing strategies for shifting precipitation patterns. Additionally, the approach democratizes access to critical soil information, particularly for resource-constrained regions, fostering sustainable development.

In summary, this research bridges the gap between advanced machine learning techniques and practical applications in agriculture and environmental management. By providing accurate, efficient, and scalable solutions, it lays the groundwork for addressing global challenges in food security, land sustainability, and climate adaptation.

References

1. Inazumi, S., Intui, S., Jotisankasa, A., Chaiprakaikeow, S., Kojima, K.: Artificial intelligence system for supporting soil classification. *Results Eng.* **8**, 100188 (2020)
2. Azmin, S.N.F.L.M., Arbaiy, N.: Soil classification based on machine learning for crop suggestion. *J. Soft Comput. Data Min.* **3**(2), 79–91 (2022)
3. Beucher, A., Møller, A.B., Greve, M.H.: Artificial neural networks and decision tree classification for predicting soil drainage classes in Denmark. *Geoderma* **352**, 351–359 (2019)
4. Rankoana, S.A.: Farmers knowledge on indigenous soil types: a review. *E-J. Hum. Arts Soc. Sci.* (2023). <https://api.semanticscholar.org/CorpusID:266514362>
5. Breiman, L.: Random forests. *Mach. Learn.* **45**, 5–32 (2001)
6. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794 (2016)
7. Ke, G., et al.: LightGBM: A highly efficient gradient boosting decision tree. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
8. Cortes, C.: Support-vector networks. *Mach. Learn.* (1995)
9. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**, 533–536 (1986)
10. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: CatBoost: unbiased boosting with categorical features. In: *Advances in Neural Information Processing Systems*, vol. 31 (2018)
11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
12. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. *Inf. Process. Manag.* **45**(4), 427–437 (2009). <https://doi.org/10.1016/j.ipm.2009.03.002>

13. Soil Climate Analysis Network (SCAN). <https://www.nrcs.usda.gov/resources/data-and-reports/soil-climate-analysis-network>. Accessed 30 Sept 2024
14. Mathew, A., Amudha, P., Sivakumari, S.: Deep learning techniques: an overview. In: *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2020*, pp. 599–608. Springer (2021)
15. Nikiforova, A.A.: Soil classification. *KO Knowl. Organ.* **46**(6), 467–488 (2019)
16. Kattenborn, T., Leitloff, J., Schiefer, F., Hinz, S.: Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. *ISPRS J. Photogrammetry Remote Sens.* **173**, 24–49 (2021)
17. Mahesh, B.: Machine learning algorithms-a review. *Int. J. Sci. Res. (IJSR)* **9**(1), 381–386 (2020)
18. Montagne, D., Cornu, S., Le Forestier, L., Cousin, I.: Soil drainage as an active agent of recent soil evolution: a review. *Pedosphere* **19**(1), 1–13 (2009)



A Comparative Analysis of FA1→3 and Lévy-Flight Based FA1→3 with Applications in Credit Card Fraud Using SMOTE Data Augmentation

Trevor Vambe¹(✉) , Mandlenkosi Gwetu² , and Serestina Viriri^{1,3}

¹ School of Mathematics, Statistics and Computer Science,
University of KwaZulu-Natal, Durban, South Africa
219095439@stu.ukzn.ac.za, viriris@ukzn.ac.za

² Department of Industrial Engineering, Stellenbosch University,
Stellenbosch, South Africa
mgwetu@sun.ac.za

³ Centre for Augmented Intelligence and Data Science, University of South Africa,
Pretoria, South Africa

Abstract. Technology in the financial sector is advancing at an accelerated rate as financial institutions seek to improve transaction speed and service convenience to customers. Several algorithms have been proposed to mitigate credit card fraud, such as swarm-based algorithms. These algorithms include, amongst others, the firefly algorithm (FA), which has gained a reputation for both efficiency and effectiveness as an optimization algorithm. There have been several variations of the algorithm, and these include, amongst others, compact Lévy-flight FA, Yin-Yang FA and FA1→3. The FA1→3 has been shown to be very effective and efficient. On the other hand, Lévy-flights have shown a lot of potential in real-life problems that are stochastic in nature. Therefore, this study proposes an algorithm to extend the FA1→3 by merging it with Lévy-flights in a bid to draw from the strengths of both existing variations. Furthermore a comparison of the three algorithms (original FA1→3, updated FA1→3 implementation and Lévy-Flight based FA1→3) is made based on experiments carried out on a credit card fraud data set. Although the original algorithm recorded the highest performance metrics, the proposed Lévy based FA1→3 algorithm proves to be more dependable.

Keywords: Lévy-flights · Algorithm · Exploration · Exploitation

1 Introduction

The prevailing success enjoyed by e-commerce is partly owed to credit card usage that has facilitated easy transacting even in the absence of funds. Unfortunately, credit card fraud (CCF) has become rampant in our societies because

of fraudsters' innovative strategies and technologies [1]. Consequently, the Federal Bureau of Investigation (FBI) received 3.26 million complaints, resulting in an estimated US\$27.6 billion in losses in the period [2018,2022] [2].

Financial institutions employ a combination of both prevention and detection strategies to protect resources, assets and customers from fraudsters [3]. Traditionally, heuristic or rule based approaches were used as preventative measures. However, there were several disadvantages associated with these approaches. For instance, they could not adapt fast enough to the dynamic nature of credit card fraud detection. In addition, they were associated with high rates of false positives. On the other hand the advent of machine learning algorithms ushered in a new era in credit card fraud detection [4]. Consequently, several machine learning and statistical algorithms have been proposed and these have been observed to perform better when datasets are balanced [5–7].

To deal with class imbalance, several data augmentation algorithms have been proposed. These include, Adaptive Synthetic Sampling (ADASYN), Random Oversampling, Synthetic Minority Oversampling (SMOTE), Generative Adversarial Networks (GANs) and their variants. The variants for SMOTE, include SMOTE-Edited Nearest Neighbour (SMOTE-ENN), Hybrid Cluster Affinity Boundary Line SMOTE (HCAB-SMOTE) and SMOTE-TOMEK [8–11]. In addition, the variants for GANs include, Deep Convolutional GAN (DCGAN), Least Squares GAN (LSGAN), Variational Autoencoder Generative Adversarial Network (VAEGAN) and Laplacian Pyramid of GAN (LAPGAN) [12–14].

There are several machine learning algorithms that have been proposed over the years and the majority of these are classification algorithms [15,16]. These machine learning algorithms include swarm based algorithms (Ant Colony Optimization, Cuckoo search, Particle Swarm Optimization, etc.) [17,18]. Swarm based systems embody the characteristics of a self organizing system that has several agents that independently interact with the environment sharing information [19].

Firefly Algorithms (FA) have proven to be both effective and efficient in parameter optimization [20]. These have several variants that include compact Lévy-flight FA, Yin-Yang Firefly and FA1→3. In addition, to the best of our knowledge the predecessors of the proposed algorithm are the Lévy-flight FA and FA1→3 and there has been no other implementation especially in credit card fraud detection.

The current research explores these firefly algorithms guided by the following questions:

1. Does a hybrid of the FA1→3 and Lévy-flight improve the algorithm's (FA1→3) exploitation and exploration capabilities?
2. Does augmenting the FA1→3 algorithm with Lévy-flight improve the performance of the algorithm (FA1→3)?

The contributions of this study are as follows:

- Proposing an alignment of the FA1→3 implementation with the actual algorithm;

- Implementing the FA1→3 algorithm in credit card fraud detection (Python) with SMOTE data augmentation;
- Integration of Lévy-Flights into FA1→3 with SMOTE data augmentation.

The remainder of this paper is organized as follows: Sect. 2 summarizes several studies that are related to the current study. In addition, Sect. 3 describes the firefly algorithm while Sect. 4 discusses Lévy flights and Sect. 5 outlines the updated FA1→3 algorithm. Section 6 presents the Lévy augmented FA1→3 algorithm flow chart. Moreover, Sect. 7 discusses the different features and characteristics of the dataset used in the experiments. Consequently, Sect. 8 explores the different experiments that were carried out. Finally, Sect. 9 summarizes the contributions of the study and highlights possible future extensions of the work.

2 Related Work

Lévy flights/walks have had extensive applications and in many of these they formed hybrids with other algorithms. Consequently, these hybrids have been tested and/or applied to several functions and problems (medical, industrial, etc.). The application areas included structural design optimization, where Lévy flights/walks were hybridized with algorithms such as Artificial Bee Colony (ABC), Invasive Weed, Slime Mold by Opposition-based Learning, Bat and Biogeography-based Optimization algorithms [21–31].

The application areas also included energy optimization, in which the proposed hybrids comprised of Lévy flight/walk and several other algorithms (Particle, Moth-flame, Jaya, Coyote, Grasshopper, Backtracking and Bat optimization algorithms) [32–44]. Additionally, Lévy flight/walk hybrids with PSO, sliding window and Whale optimization algorithm have been applied in job scheduling, electronic communications and chaotic long range interacting systems [45–48]. Furthermore, some researchers have used hybrids of Lévy flights/walks with Multi-verse, Chaotic Atom Search, Reptile Search or Simulated Annealing Algorithms in electronic design optimizations [49–52]. Also, Lévy flights/walks have had several other application areas such as communication, boiler combustion optimization, parameter optimization in automatic carrier landing systems and Process Identifier (PID) controller optimization where Lévy flights were utilized in conjunction with algorithms such as Vortex Search, NelderMead, Pigeon-Inspired optimization algorithms [47, 48, 53–55].

The work done by Wang et al. [56] provides a non-exhaustive list of variants of the firefly algorithm (2010–2019). Most of the work proposed hybrids of the firefly algorithm with several other algorithms [56–58]. The literature review focused on the strategies and any modified aspects of the firefly algorithm. Consequently, the Authors proposed a model based on Cauchy mutation [56].

Table 1 provides a non-exhaustive list of hybrids of the firefly algorithm and some of its application areas. There has been a lot of research that used a conjunction of Lévy-flights and the firefly algorithm. The application areas ranging from bench-marking datasets such as CEC2014 and CEC2020, image segmentation to several engineering problems. It was also evident that functions such

Table 1. A Table showing the different hybrids of the firefly algorithm and the respective application areas

Papers	Algorithm or Distributions forming hybrid with the Firefly Algorithm	Application Areas firefly algorithm
[57,59–87]	Lévy flights	Appendicitis, Australian credit approval, Breast cancer, BUPA liver disorder, clustering in wireless sensors, German credit data, Haberman surgery survival, Image segmentation, Intensity modulated radiation therapy, Inter-vehicle communication, Parkinson, PI controller, machine scheduling, PID controller, PIMA Indian diabetes, Power flow problems (thermal and wind), Robotics, Satelliteimage enhancement, Sentimental Analysis. Several engineering problems such as a three-bar truss design, Cantilever beam, corrugated bulkhead design, pressure vessel design, Welded beam design, tabular column design. Bench-marking functions and datasets such as CEC2014 and CEC2020,
[58,59,64]	Gaussian Distribution	Functions: Several bench marking functions such as Ackey and De Jong's sphere.
[70,77,88]	or Gauss Map	Functions: Several bench marking functions such Rastrigin and Rosenbrock.
[70,89–91]	Chaotic Map	Image segmentation, Power flow problems (thermal and wind), Functions: Several bench marking functions such 2 ⁿ Minima and Ackley.
[56,92,93]	Cauchy Distribution	Image enhancement Functions: Several bench marking functions such Ackley and Bukin.
[94–96]	Differential Evolution	Functions: Several bench marking functions such Ackley and Bohachevsky2. steganography and several engineering problems.
[59,70]	Logistic map	Functions: Several bench marking functions such 2 ⁿ Minima and Ackey.
[64,87]	Brownian Motion	Image segmentation, Welded beam problems
[63]	Simulated Annealing	Appendicitis, Australian Credit approaval, Breast cancer, Bupa Liver, Disorders Fourclass, German Credit data, Haberman Surgery Survival, Parkinsons PIMA Indian diabetes and Statlog.
[97]	Genetic algorithm, Ant Colony Optimization	Functions: Several bench marking functions such Sphere and Ackley.
[67]	Eagle Strategy	Functions: Several bench marking functions such Ackley and De Jong.
[70]	Uniform distribution, RSiTFC	Functions: Several bench marking functions such Ackley and De Jong's sphere.
[71]	Fish Swarm Algorithm	Functions: Several bench marking functions such Ackley and Dixon and price.
[75,85]	Multilayer Perceptron	Sentimental Analysis (Web reviews)
[87]	Leader follower population model	Functions: Several bench marking functions.
[98]	hierarchical strategy	Functions: Several bench marking functions. Several real world engineering problems.

as sphere, Ackley and Griewank were functions that were commonly used in evaluating proposed algorithms [70, 71, 89–91, 97].

There are several algorithms that have been proposed for credit card fraud detection and these can be categorized into anomaly and misuse detection. The former utilizes supervised while the later uses unsupervised machine learning algorithms. There are several supervised machine learning algorithms that have been proposed over the years. Some of these include but are not limited to Artificial Immune Systems (AIS), Support Vector Machines (SVMs), Neural Networks (NNs), Naive Bayes (NBs), Generative Adversarial Networks (GANs), Logistic Regression (LR), Rule Induction (RI) and Decision Trees (DTs) [1, 5, 7, 12, 99–117]. On the other hand, there are several unsupervised machine learning algorithms that have been proposed. Some of these include Self Organizing Maps (SOM), K-means, Hidden Markov Model Method, etc. [1, 118].

Additionally, there are several hybrids that have been proposed for credit card fraud detection [1, 104, 119, 120]. In addition, the work done by Banerjee et al. [121] and several other Authors compared the performance of a plethora of algorithms on credit card fraud detection [15, 100, 106, 121–123]. Furthermore, there are several studies that have utilized deep learning for CCFD. Consequently, several comparative analyses of diverse deep learning algorithms have been carried out by a plethora of researchers [122, 124–129].

Also, there are several studies that have utilized GANs whilst others have formulated CCFD as a sequence problem [130, 131]. Furthermore, other algorithms were hybrids of supervised and unsupervised machine learning algorithms [132]. On the other hand, some proposed algorithms were based on ensembles [118, 133–138], hybrids of machine learning and big data analytics [139], transferred learning [140], data analytics [141, 142], data mining [143–148], consumer incentives [149] and sliding windows algorithms [138, 150].

3 The Firefly Algorithm

There are several parameter optimization algorithms that have been proposed that are inspired by the intelligence exhibited by biological systems (Swarm Intelligence). These swarms range from schools of fish, bacterial growth to particles [151–154]. The swarm agents (non-sophisticated) individually interact with the environment sharing information and thereby creating complex systems [19, 153, 154]. These systems are best candidates for parameter optimization in a dynamic environment as they are self learning and adaptive in nature [19, 155–157]. Additionally, they have strong exploitation and exploration capabilities [17, 153, 154, 158].

Fireflies are some of the most effective and efficient swarm based optimization algorithms. They have proven themselves especially in problems that are high dimensional, complex, multi-modal and/or nonlinear [158]. The algorithm and its variants have the following three (3) idealized rules [57, 60, 65, 89, 153, 154, 159, 160]:

1. All fireflies are bisexual and hence attraction is not limited by gender.

2. The attractiveness of each firefly is taken to be proportional to their light intensity and inversely proportional to the square of the distance between the two fireflies. The firefly with the smaller intensity value moves towards the one with a higher intensity value.
3. Light intensity is determined by the objective function.

Fireflies have a distinctive flashing light that researchers have postulated to be for mating, warding off predators, e.t.c [89,152–154,160]. Moreover, light intensity (I) is known to be inversely proportional to the square of distance (r^2) from the firefly. Consequently, the relationship between light intensity and distance from the firefly is as shown in Eq. 1 [20,65,89,153,161]:

$$I \propto \frac{1}{r^2}. \quad (1)$$

where r is distance from the firefly and I is the light intensity at the point. Furthermore, Gaussian approximation can be used in simplifying Eq. 1 to Eq. 2 [20,65,89,153,161]:

$$I(r) = I_0 e^{-\gamma r^2}. \quad (2)$$

where I and r are as discussed above. In addition, I_0 is the intensity at the firefly while γ is the light absorption coefficient. Also, it is believed that a firefly's (F_1) attraction (β) to another firefly (F_2) is proportional to the light intensity of F_2 and this is mathematically represented as shown in Eq. 3 [20,65,89,153,161]:

$$\beta(r) = \beta_0 e^{-\gamma r^2}. \quad (3)$$

where r , γ and r are as discussed above. Moreover, β is the attraction of the firefly towards another firefly at distance r and β_0 is the attraction when the two fireflies are at the same location. Additionally, if Eq. 3 holds true it implies Eq. 4 holds true [20,65,89,153,161]:

$$\beta(r_{ij}) = \beta_0 e^{-\gamma r_{ij}^2}. \quad (4)$$

where the two fireflies are at positions i and j (i is not the same as j) and the rest of the parameters are as discussed above. Additionally, the distance between the two location is given by Eq. 5 [20,65,89,153,161]:

$$r_{ij} = ||x_i^t - x_j^t|| = \sqrt{\sum_{m=1}^M (x_{im}^t - x_{jm}^t)^2}, i \neq j. \quad (5)$$

where r_{ij} is the distance between positions i and j in the search space. Moreover, t is the time stamp and M is the number of dimensions of the search space. In addition x_{im}^t is the distance from the origin to position i parallel to the m^{th} dimension at time t .

Furthermore, the firefly algorithm states that the firefly (F_1) with lower light intensity moves towards the other firefly (F_2). Moreover, the new position is updated using Eq. 6 [17,20,89,162]:

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r^2} (x_j^t - x_i^t) + \alpha \sigma_i. \quad (6)$$

where x_i^{t+1} and x_i^t are the future and current positions respectively. In addition, B_0 , γ , r and x_j^t are as discussed above. Furthermore, α is a random variable in the range $[0, 1]$ and σ_i is a vector whose scalar components are generated from a Gaussian distribution.

In cases where the intensities of the two fireflies are the same, one of the fireflies is moved randomly within the search space [161]. Several Authors have proposed a plethora of variants of the firefly algorithm. For instance, there are several hybrids as indicated in Sect. 2. On the other hand, Authors like Kazemzadeh [163] proposed a variant that introduced new ways of handling memory, integration of a few genetic algorithm features and an updated update equation (Eq. 6) [163]. Additionally, the standard firefly algorithm's parameters are traditionally set manually. However, some Authors have researched on optimization strategies for the parameters as they control the algorithm's exploitation and exploration capabilities [58, 164–168]. On the other hand, some researchers have researched on improvements to the strategies implemented by the algorithm. For instance, the attraction component or the update equation in general [169].

The study conducted by Ghasemi et al. proposed an update equation entitled FA1→3 and the equation was as shown in Eq. 7 [20]:

$$\begin{aligned}
 x_i^{new} = x_i + & \\
 & \text{sign}(F(x_i) - F(x_{j1})) * \beta_1 e^{-\gamma r_{ij1}^2} * (x_{j1} - x_i) + \\
 & \text{sign}(F(x_i) - F(x_{j2})) * \beta_2 e^{-\gamma r_{ij2}^2} * (x_{j2} - x_i) + \\
 & \text{sign}(F(x_i) - F(x_{j3})) * \beta_3 e^{-\gamma r_{ij3}^2} * (x_{j3} - x_i) + \\
 & \alpha \epsilon_i.
 \end{aligned} \tag{7}$$

where x_i^{new} and x_i are the future and current positions respectively. In addition, the function $\text{sign}()$ returns values -1, 1 and 0 when $F(x_i) > F(x_j)$, $F(x_i) < F(x_j)$ and $F(x_i) = F(x_j)$ respectively. In addition, $F(x)$ evaluates location x 's objective function value. Moreover, β_1 represents the attraction towards the firefly at location x_{j1} if the two fireflies are at the same location. This also applies to β_2 and β_3 and positions x_{j2} and x_{j3} respectively. Furthermore, the attraction of the firefly located at position x_i towards another at each of the positions x_{j1} , x_{j2} and x_{j3} is given by $\beta_1 e^{-\gamma r_{ij1}^2}$, $\beta_2 e^{-\gamma r_{ij2}^2}$ and $\beta_3 e^{-\gamma r_{ij3}^2}$ respectively. Moreover, $(x_{j1} - x_i)$ provides a vector whose scalar components indicate the distance between x_i and x_{j1} parallel to each of the dimensions of the search space. This also applies to $(x_{j2} - x_i)$ and $(x_{j3} - x_i)$. Additionally, the term $\alpha \epsilon_i$ is a randomization factor with α and ϵ_i being exactly as discussed in Sect. 3. Furthermore, the current study seeks to extend this study. Additionally, the following section covers the theory behind Lévy flights.

4 The Lévy Flights

Random walks are defined as stochastic trajectories of waves or particles. Furthermore, Brownian motion (fluid dynamics) research was the pioneering study

on random walks [170,171]. Additionally, in the period preceding 1930, Brownian motions were extensively studied by Einstein, Langevin, Wiener and several others. Ensuing from this, there was wide spread research in both Brownian motion and Lévy processes [171–173].

A Stochastic Markov process (Θ_t) is defined in some real dimensional space $\mathbb{R}^n$, with order n . Additionally, the process Θ_t is defined as a group of related random variables that share common characteristics. Furthermore, the parameter t ($0 \leq t \leq \infty$) has a probability space (ζ, χ, Φ) . Where, ζ , χ and Φ is firstly, the universal set of outcomes, secondly, $\forall \Psi \subset \zeta : \Psi \neq \emptyset$ it then follows that $\mathbb{A}, \mathbb{B} \subset \zeta \Rightarrow \mathbb{A} \cup \mathbb{B} \subset \zeta$ and $\mathbb{A} \cap \mathbb{B} \subset \zeta$, and thirdly the probability of Θ_t on χ respectively. Additionally, a Stochastic Markov process possessing the following additional characteristics is defined as a Lévy process [170,172,174]:

- i For the sequence $\{\Theta_t : t \in \{0, \mathbb{N}\}\}$, where $\{t_i < t_j, \forall j > i\}$, all increments ($\epsilon = \Theta_{t+1} - \Theta_t$) are independent.
- ii Θ_t is defined for $t = 0$.
- iii The distribution of $\{\Theta_{t+\eta} - \Theta_t : t \in \{0, \mathbb{N}\}\}$ is independent (time homogeneous) of the value of η .
- iv Θ_t is continuous with respect to time (t).
- v With respect to time (t), Θ_t has a left limit and is right tailed.

Additionally, the generation of random variables that follow Lévy flights primarily comprise of two steps, that is, the way a random bearing is selected and an algorithm for observing a particular Lévy distribution. Furthermore, Lévy processes can be mathematically expressed as shown in Eq. 8 [175,176]:

$$L(\eta, \omega, \lambda) = \begin{cases} \sqrt{\phi} \exp[\rho\tau], & 0 < \mu < \eta < +\infty, \\ 0, & \eta \leq 0. \end{cases} \quad (8)$$

where, ω and λ are Lévy distribution and search space navigation scale factors respectively. Moreover, $\rho = -\omega/2(\eta - \lambda)$, $\tau = 1/(\eta - \lambda)^{\frac{3}{2}}$ and $\phi = \omega/2\pi$ [175]. Further more, Lévy processes are known to exhibit several characteristics such as, infinite mean and variance, and generalization to the Central value theorem [174,177–179].

5 Proposed Update Equation

Several processes such as physical phenomena, biological and financial processes could be modeled as Lévy distributions [20,171,180]. Furthermore, Levy flights' core principle is random walks and this forms the central idea for both exploitation and exploration [181]. Consequently, levy flights have been observed to provide a reasonable trade-off between exploitation and exploration [182]. Additionally, the properties of levy flights helps avoid algorithms being stuck in local minimum and also improve the convergence rate of the given algorithm

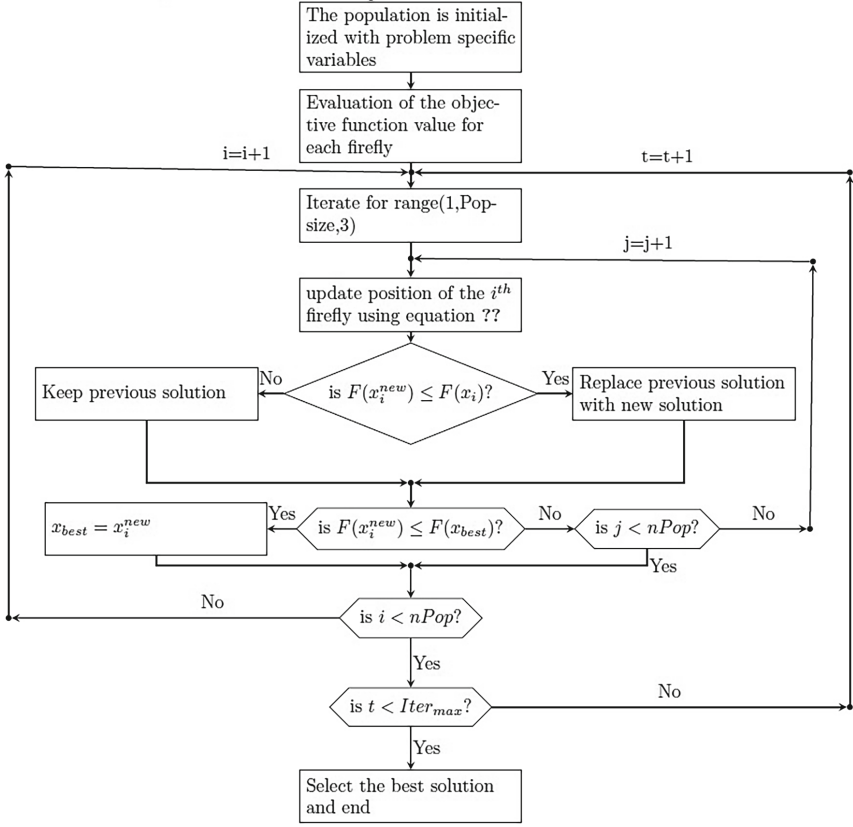


Fig. 1. Lévy augmented $FA1 \rightarrow 3$ Algorithm flow chart

[182]. Consequently, the proposed algorithm (Eq. 9) is a hybrid of the $FA1 \rightarrow 3$ algorithm [20] and Lévy flights [71, 85, 176]:

$$\begin{aligned}
 x_i^{ftr} = & x_i \\
 & + \varphi(\Pi(x_i, x_{j1})) * v_1^{r_{ij1}} * \Xi(x_{j1}, x_i) + \varphi(\Pi(x_i, x_{j2})) * v_2^{r_{ij2}} * \Xi(x_{j2}, x_i) \\
 & + \varphi(\Pi(x_i, x_{j3})) * v_3^{r_{ij3}} * \Xi(x_{j3}, x_i) + \alpha \varphi([rand(0, 1) - 0.5]) \otimes Lévy. \quad (9)
 \end{aligned}$$

Moreover, the algorithm has been implemented in credit card fraud detection. Furthermore, the proposed algorithm uses an update rule that has x_i and x_i^{ftr} as current and future positions within the search space, respectively. In addition, the representation of the update rule has been simplified by defining the functions such as Eq. 10:

$$\varphi() = np.sign(\omega). \quad (10)$$

where the function $np.sign()$ is a Python function that returns -1, 1, and 0 for the $\mathbb{Z}(x_i) < \mathbb{Z}(x_{ja})$, $\mathbb{Z}(x_i) > \mathbb{Z}(x_{ja})$ and $\mathbb{Z}(x_i) = \mathbb{Z}(x_{ja})$ respectively. Moreover, ω is

a variable representing the output of Eq. 12 and in some cases ($rand(0, 1) - 0.5$). Additionally, the function in Eq. 11 is defined:

$$\Pi(x_i, x_{ja}) = \mathbb{Z}(x_i) - \mathbb{Z}(x_{ja}). \quad (11)$$

where $\mathbb{Z}()$ is the objective function and the function $\Pi(x_i, x_{ja})$ evaluates the difference between the objective function values of the locations x_i and x_{ja} in the search space. Furthermore, the function in Eq. 12 is defined:

$$\Xi(x_{ja}, x_i) = x_{ja} - x_i. \quad (12)$$

where x_{ja} and x_i are positions in the search space. Addition, the function $\Xi(x_{ja}, x_i)$ returns the displacement vector from x_i to x_{ja} . Furthermore, the function in Eq. 13 is defined:

$$v_b = \beta_b e^{-\gamma}. \quad (13)$$

where the parameters, γ and α are the light absorption coefficient and randomization parameter, respectively. Additionally, a and b are constants defined in the range $[1, 2, 3]$. Consequently, the proposed update equation is as shown in Eq. 9. In which, the function $rand(0, 1)$ generates random number in the range $[0, 1]$. As a result, the component $\varphi([rand(0, 1) - 0.5])$ provides a random direction for the firefly (sign).

On the other hand, the operator $\otimes$ facilitates element-wise multiplication. Furthermore, the element-wise size of the firefly's movement is derived from a Lévy distribution. Additionally, Eq. 9 can be broadly broken down into the following three parts:

1. x_i ,
2. $\varphi(\Pi(x_i, x_{j1})) * v_1^{r_{ij1}} * \Xi(x_{j1}, x_i) +$
 $\varphi(\Pi(x_i, x_{j2})) * v_2^{r_{ij2}} * \Xi(x_{j2}, x_i) +$
 $\varphi(\Pi(x_i, x_{j3})) * v_3^{r_{ij3}} * \Xi(x_{j3}, x_i),$
3. $\alpha\varphi([rand(0, 1) - 0.5]) \otimes \text{Lévy}.$

Where the first part is the current location of the firefly within the search space. Additionally, fireflies are ranked based on objective function value (Best to least). Then the second part provides an attraction component influenced by the three fireflies that come after the current firefly in the objective function based ranking. Additionally, the last component provides an added randomization component based on Lévy flights.

6 Lévy Augmented $FA1 \rightarrow 3$ Algorithm Flow Chart

The Lévy augmented $FA1 \rightarrow 3$ algorithm (Fig. 1) begins with the generation of a population of fireflies. Where each firefly has identical problem specific random variables. Additionally, the algorithm initializes variables of each member of the population. There after, a population ranked list [$best \rightarrow worst$] is generated using the objective function. Then, the algorithm iterates (*loopi*) over the ranked

list using Eq. 9 in proposing position updates for each firefly. Furthermore, each firefly's position is set as an optimal value selected from the set $\{\mathbb{Z}(x_i), \mathbb{Z}(x_i^{ftr})\}$, where $\mathbb{Z}$, x_i and x_i^{ftr} are as discussed in Sect. 5. Additionally, the population best results are also updated as an optimal solution selected from the set $\{\mathbb{Z}(x_i^{updated}), \mathbb{Z}(x^{best})\}$, where $x_i^{updated}$ is the updated position of each firefly while x^{best} is the current recorded optimal solution within the search space. Moreover, the algorithm is design to repeat (*loopt*) several times (user defined repetitions). The following section summarizes the dataset that was used during the experiments.

7 Kaggle Credit Card Fraud Detection Dataset

The current study used the Kaggle credit fraud detection dataset, which comprised of transaction recorded for European credit cardholders. The dataset has two hundred and eighty-four thousand eight hundred and seven (284 807) transactions captured over a period of two (2) days in the thirteenth year of the twenty first century (2013). Furthermore, four hundred and ninety-two (492) of the recorded transactions were fraudulent while the rest were non-fraudulent. In addition, the dataset was pre-processed such that it would be difficult for anyone to track personal details of those individuals involved in the transactions. This mitigated the dangers of fraud to those whose transactions were recorded in the dataset. Consequently, the majority of the features were transformed using Principle Component Analysis (PCA) except for “Amount” and “time”. Also, the different feature names were converted to $V_1, V_2, \dots, V_{28}$. On the other hand “Amount” and “class” remained unchanged [183]. Moreover, the following section covers the experimental work carried out during the study.

8 Experimental Work

The work carried out by Ghasemi et al. [20] proposed a variant of the firefly algorithm ($FA1 \rightarrow 3$) which they implemented in MATLAB. Additionally, the algorithm was used in solving engineering problems [20]. On the other hand, the current research implemented the algorithm ($FA1 \rightarrow 3$) in Python and applied it to credit card fraud detection.

There were three experiments conducted in this study. The first experiment implemented the algorithm mimicking its implementation in the work carried out by Ghasemi et al. [20]. While the second proposed an implementation that the Authors felt better aligned with the algorithm ($FA1 \rightarrow 3$). Finally, the Authors proposed a Lévy augmented $FA1 \rightarrow 3$ algorithm. Furthermore, each of the three experiments had two sets of data collected:

- i The optimal solutions for each of the five runs (each run having several iterations).
- ii A collection of all the values generated for all the parameters.

The following subsection discusses the experimental results associated the implementation that mimicked the one done by Ghasemi et al. [20].

8.1 Original Algorithm Implementation

The researchers implemented the algorithm proposed by Ghasemi et al. in python. This was done by converting the original code line by line to python. In addition, any available functions were also redefined in python. The python version of the algorithm was then proposed for use CCFD. In addition, this algorithm exhibit several properties that include fireflies exploring and exploiting the multi-dimensional search space with each iteration. Furthermore, the algorithm keeps track of the best position recorded up to any point during the iteration. Eventually, the best position for each iteration is recorded.

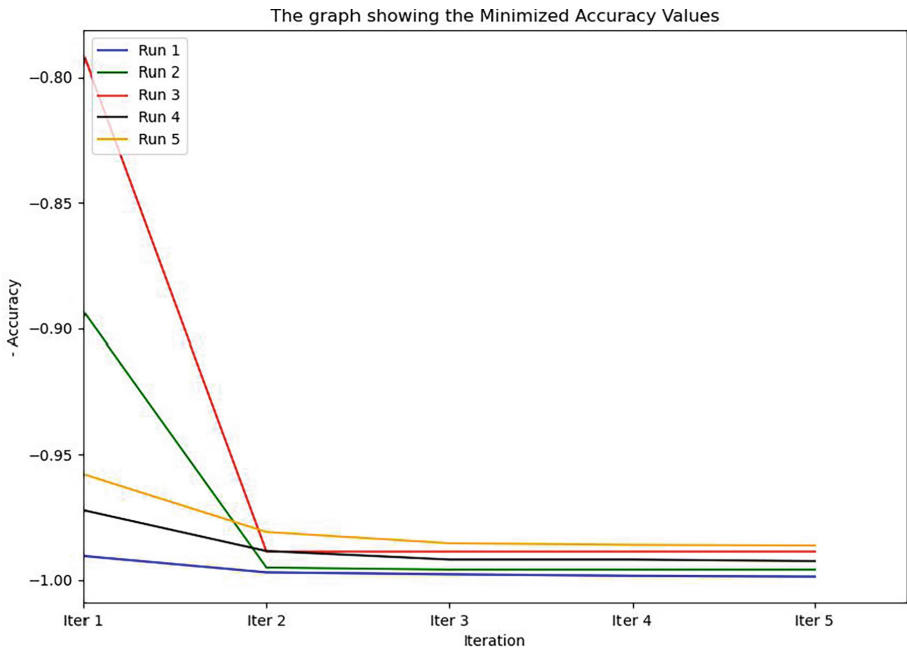


Fig. 2. Graph showing the results of the original algorithm for five runs

Each experiment comprised of five runs with each having five iterations. Furthermore, the best position for each iteration was recorded for the five runs. Consequently, the corresponding accuracy values for each best position was recorded and were as depicted in Fig. 2.

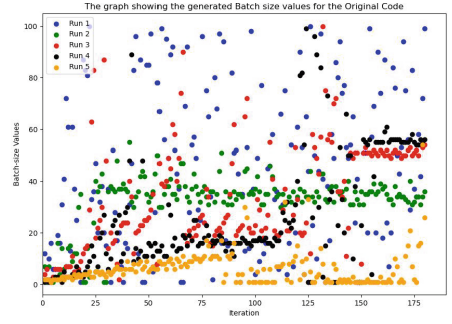
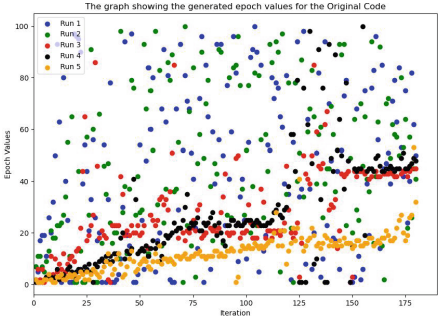


Fig. 3. Graphs showing the distribution of values generated by the original code.

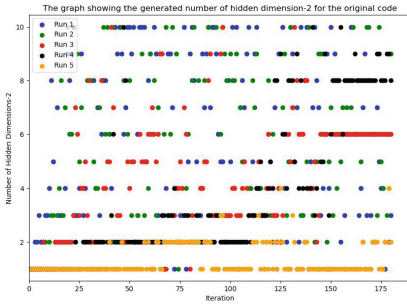


Fig. 4. Graph showing values for hidden dimensions-2 for the original code

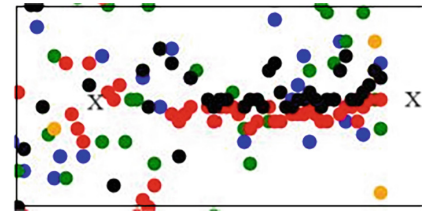


Fig. 5. Num-epochs:zoomed exploited area

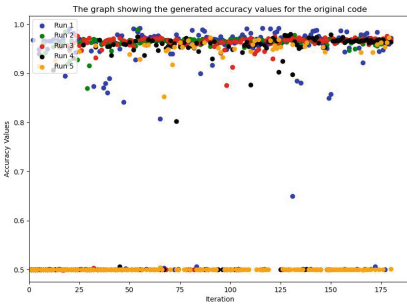


Fig. 6. Accuracy values for the original code.

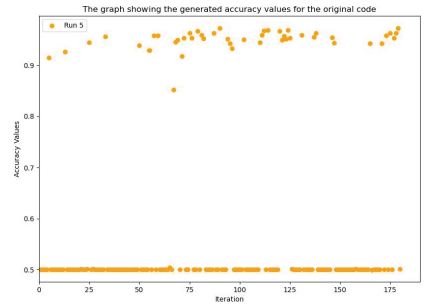


Fig. 7. Accuracy values (Run 5) for the original code.

In Fig. 2, all the runs show low initial values of accuracy that increase with an increase in the modulus of the iteration number. The third run had the lowest recorded accuracy values and one of the lowest recorded accuracy values (less than 80%). In addition, the second run recorded an accuracy value of less than

90%. On the other hand, the other three runs had accuracy values that were almost the same and these were in the range $[-0.9579, -0.9986]$. Above all, the majority of the accuracy values for all the runs were in the range $(-0.9808, -0.9986)$.

Plots of Values Generated by the Algorithm. The firefly algorithm or any of its derivatives would best serve its purposes if it were to be strong at both exploitation and exploration. Consequently, all the values generated for any of the parameters for all the iterations and runs were collected and plotted as shown in Fig. 3 and 4. Where the graphs for encoding dimensions, momentum, learning rate and alpha are similar to those for number of epochs and batch size. In addition, the graph for hidden dimension 2 is similar to the one for hidden dimensions. In addition, for all the parameters it is evident that the algorithm initially generates random values. There after, the generated values can be approximated by a linear function. In some cases the linear function would be horizontal in nature in some cases it would be a linear function with a positive gradient. In some graphs there is a combination of both horizontal and positive gradient functions. These graphs indicate that the generated values strongly depend on the initial randomly generated value. In this case the algorithm is strong in exploitation and weak in exploration.

For the number of epochs the points could be approximated by a linear function with a positive gradient. It can also be observed that the algorithm generated values that covered the whole search space in each run. However, the majority of the values generated were centered around the initially generated values. It therefore follows that the implementation allowed for exploration but mainly focused on exploiting the area around the initial value. Also, it mainly focused on the values above the initial value (due to the positive gradient). These characteristics seem to be common in all parameters other than hidden-dimension 1 and 2.

There is only one place where the algorithm seems to be exploiting. Figure 5 showed the stated area. However a closer look at the graph shows that this is an extension of the linear function even though there seems to be a break in the flow of points on the region around the linear region. In Fig. 5 it can be observed that the algorithm is exploiting the region around the marked area. This seems to be a common occurrence for all parameters especially in the third run. Furthermore, in hidden-dimension 1 and 2 the generated data seem to follow horizontal functions that are mainly centered around the initial value. That is it maintains the initial value through out the iterations with chances of an increase by one as the iterations continue. However there are some runs where the values were spread out over a wider range. An example being for hidden-dimension 2, values for run 3 seems to be spread over a wider region. The hypothetical line connecting the two X s would form a linear line that the generated points are exploiting (black dots). The same would apply for red dots. In the image there is a region being exploited.

Plots Accuracy Values. As the algorithm was going through its different attempts the accuracy (Fig. 6) values were recorded. Further more observing the points one would note that for each parameter the accuracy were mainly evenly distributed in regions just below 1 for most of the runs. However for run 5 there were mainly centered around 0.5 (Fig. 7). Consequently, a close look at the graphs indicate that run 5 was associated with low values for all the parameters.

8.2 Proposed Updates to the Original Algorithm

The code in Eqs. 14–20:

$$rij = norm(pop(i).Position - pop(j).Po..., \quad (14)$$

$$beta = beta0 * exp(-gamma * rij^m), \quad (15)$$

$$e = delta * unifrnd(-1, +1, VarSize), \quad (16)$$

$$rij2 = norm(pop(i).Position - pop(j + 1).Pos..., \quad (17)$$

$$beta2 = beta0 * exp(-gamma * rij2^m), \quad (18)$$

$$rij3 = norm(pop(i).Position - pop(j + 2).Pos..., \quad (19)$$

$$beta3 = beta0 * exp(-gamma * rij3^m), \quad (20)$$

shows part of the MATLAB implementation of the update rule (Eq. 21):

$$\begin{aligned} x_i^{new} = & x_i + sign(F(x_i) - F(x_{j1}))(x_{j1} - x_i) + sign(F(x_i) - F(x_{j2}))(x_{j2} - x_i) \\ & + sign(F(x_i) - F(x_{j3}))(x_{j3} - x_i) + \alpha e_i. \end{aligned} \quad (21)$$

Additionally, it was observed that the algorithm was not implemented in a way that completely aligned with the algorithm (Eq. 21). Particularly, Eqs. 17, 18 and 19 were not utilized in the code implementation. Consequently, the current Authors proposed an update of the code in Python (Eq. 22):

$$\begin{aligned} newSol["Position"] = & (np.array([float(string) for string in \\ & pop[i]["Position"]])) + beta1 * a1 * (np.array([float(string) \\ & for string in pop[j]["Position"]])) - np.array([float(string) \\ & for string in pop[i]["Position"]])) + beta2 * a2 * \\ & (np.array([float(string) for string in pop[j + 1]["Position"]])) \\ & - np.array([float(string) for string in pop[i]["Position"]])) + \\ & beta3 * a3 * (np.array([float(string) for string in \\ & pop[j + 2]["Position"]])) - np.array([float(string) \\ & for string in pop[i]["Position"]])) + alpha * e). \end{aligned} \quad (22)$$

where beta1 is obtained using Eq. 23:

$$beta1 = er1 * beta0 * math.exp(-gamma * (rij ** m)). \quad (23)$$

In addition, r_{ij} was implemented as shown in Eq. 24:

$$rij = \text{norm}(\text{np.array}([\text{float}(\text{string}) \text{ for string in } \text{new_op}[i][\text{"Position"}]]) - \text{np.array}([\text{float}(\text{string}) \text{ for string in } \text{pop}[j][\text{"Position"}]])) \quad (24)$$

where, Eqs. 23 and 24 demonstrated how the exponential component was included in Eq. 22. Additionally, the following section covers the discussion on the experiments carried out using the updated implementation.

8.3 Updated Original Algorithm Implementation

The Authors implemented the updated algorithm in python. Experiments that are similar to those in Sect. 8.1 were carried out for the updated implementation. Furthermore, the accuracy values that were associated with each best recorded value were as shown in Fig. 8. Moreover, the best accuracy values were in the range $[-0.9843, -0.9984]$, with the initial iterations having better accuracy values as opposed to the last two runs.

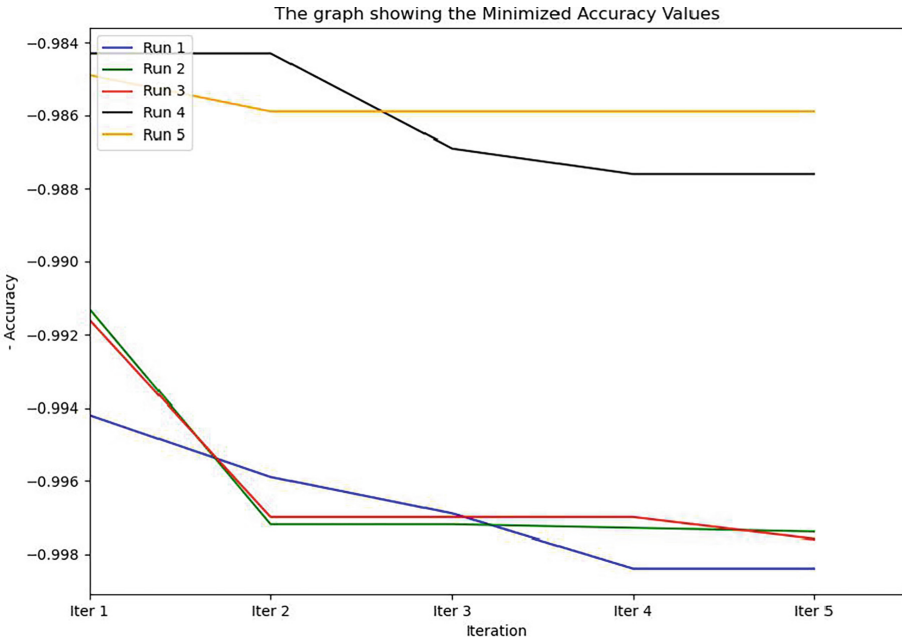


Fig. 8. The graph showing best accuracy values for the algorithm

Additionally, it was also evident that the forth and fifth runs had the worst results, whereas the results for rest of the runs slightly higher and comparable.

Moreover, it could be observed that the best result in the original implementation was better than that of the updated implementation. It was also observed that the initial iterations had accuracy values that were lower than the last iterations.

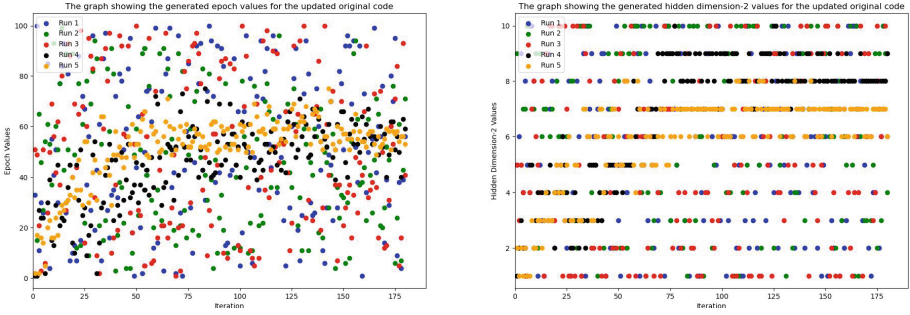


Fig. 9. The graphs showing values generated by the updated algorithm

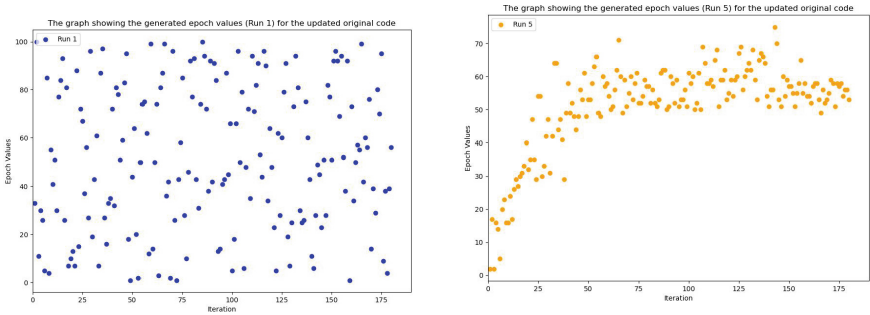


Fig. 10. Epoch values generated in Run 1

Fig. 11. Epoch values generated in Run 5

Plots of Values Generated by the Algorithm. The individual points were also plotted and observed to see how well the algorithm explored and exploited the search area. The graphs in Fig. 9 show the distribution of the various points generated by the algorithm. Where the graph for epochs is similar to the graph for batch size, encoding dimensions, and other parameters except for hidden dimension 1 and 2. On the other hand, the graph for hidden dimension 2 was similar to the one for hidden dimension 1. In addition, it was observed that the values generated by the algorithm were covering the whole search space (exploration) mainly for runs 1, 2 and 3. Figure 10 shows the distribution of point generated for number of epochs in run 1. Furthermore, there are several areas in the search space that has clusters of dots. These areas would indicate areas in which the algorithm was exploiting.

Additionally, the distribution of the generated points in runs 4 and 5 could be approximated by Eq. 25:

$$y = A\sqrt{Bx}. \quad (25)$$

where A and B are constants. Furthermore, Fig. 11 shows the distribution of points generated for number of epochs in run 5. This trend was similar for all parameters for all runs. Moreover, the algorithm in this instance increments the values and seems to be asymptotic to some value. In Fig. 11 the function that could approximate the generated points is asymptotic to 70%. However, the algorithm is exploiting the region in the range [50%, 60%].

8.4 Plots for Accuracy Values for the Updated Algorithm

The accuracy (Fig. 12) values around 0.5 were significantly less than those for the original algorithm (Fig. 6). There is a tight range of values for the updated algorithm as opposed to the original algorithm.

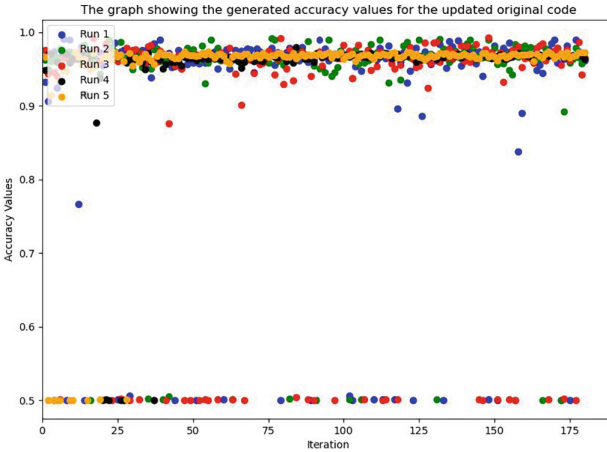


Fig. 12. Graph showing values for accuracy for the updated code

8.5 Lévy Distribution Based Results

The Authors used a Lévy based FA1→3 in an additional set of experiments. The experiments were exactly the same as those carried out in Sects. 8.1 and 8.2.

The graph in Fig. 13 shows that the best accuracy values for the various runs and iterations were in the rage $[-0.9943, -0.9981]$. In addition, the algorithm produced better results in the last two runs. Furthermore, the original code produced the highest best accuracy value but the Lévy based algorithm had the tightest range of values. In addition, the results improved with more runs.

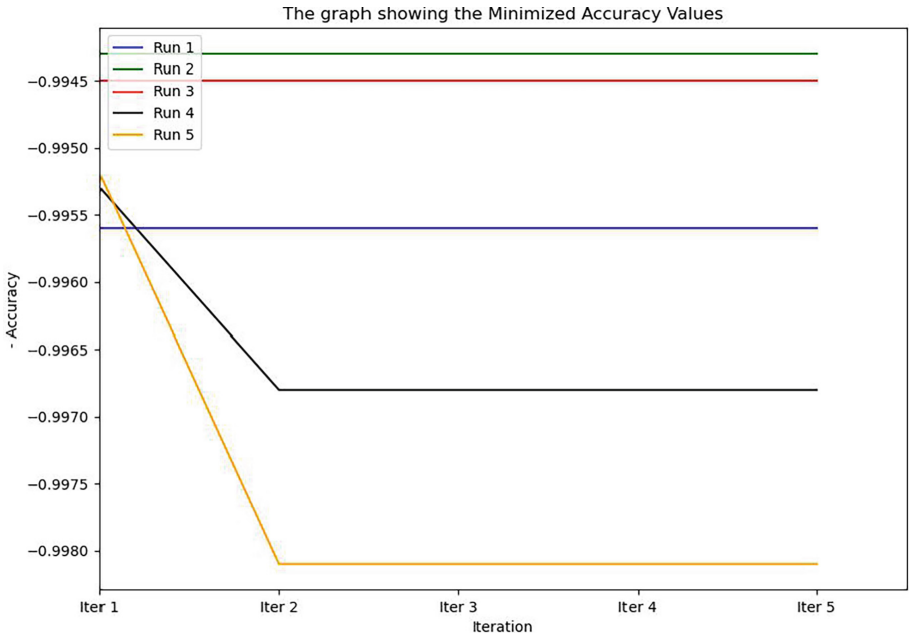


Fig. 13. The graph showing best accuracy values for the Lévy based algorithm

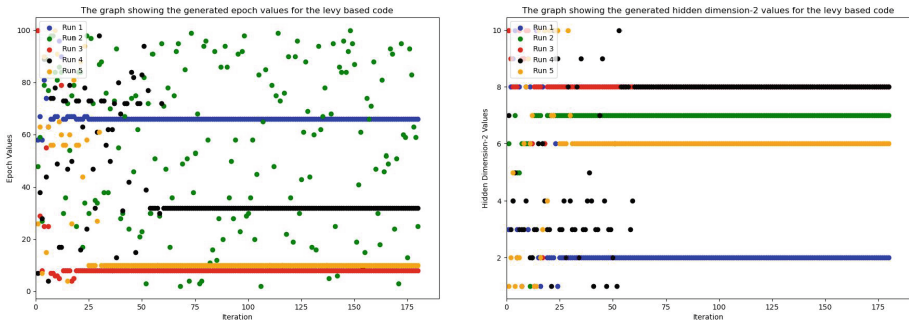


Fig. 14. The graph showing the generated values for each parameter

The Lévy based FA1→3 generated values for each parameter were recorded to observe how well the algorithm explored and exploited its search space. Additionally, the graphs in Fig. 14 shows the plotted values for the different parameters for the different runs and iterations. Where the graph for epochs is similar to the graph for several other graphs as stated in the other two experiments. In addition, the graph for hidden dimension 2 is similar to the graph for hidden dimension 1.

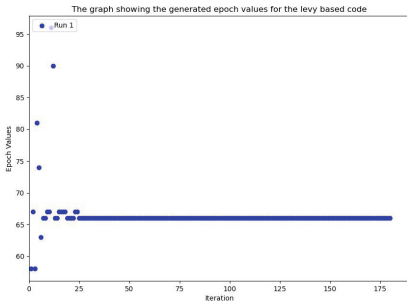


Fig. 15. Values generated for epochs in run 1

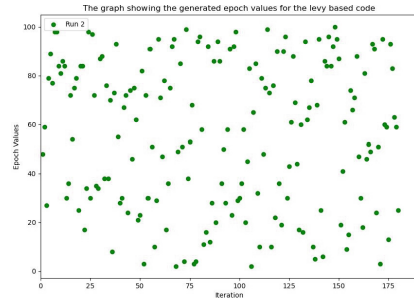


Fig. 16. Values generated for epochs in run 2

In all the graphs in Fig. 14 for most runs: the first points explored the search space. However, the algorithm gets stuck in one of the values. Consequently, it was observed that the algorithm eventually started holding one parameter constant while exploring the rest. Additionally, the number of parameters held constant continued to increase proportional to the number of iterations. This persisted until all the values for all the parameters were constant. Also, the different runs explored and exploited several values within the search space. Furthermore, typical examples of what was discussed above were the generated epoch values for runs 1, 2 and 4 shown in Fig. 15, 16 and 17 respectively.

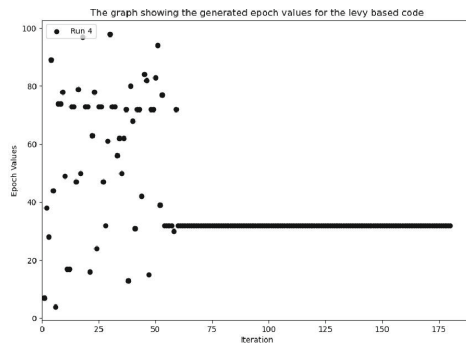


Fig. 17. Values generated for epochs in run 4

In the example shown in Fig. 15 the algorithm explored a few values in the range (58, 100) and thereafter epoch values for the run were observed to assume a constant value of 66. Also, it could be generalized that there was very little of both search space exploration and exploitation done for the parameter in this run. On the other hand, the diagram in Fig. 16 showed the generated epoch values for run 2.

In the diagram in Fig. 16, the algorithm generated values that covered the whole search space. Additionally, it was observe that any horizontal line drawn will cut at least one dot with a very high probability. This showed that the values were explored but in some cases in different iterations. Additionally, scrutinizing the graph showed that the concentration of dots varied from one area to another with some areas having no dots. Furthermore, areas with no dots were neither explore nor exploited in the respective iterations. Additionally, areas with dots were explored and the concentration of dots showed the level of exploitation in the respective iterations. On the other hand Fig. 17 depicted the generated epoch values for run 4.

In Fig. 17 the graph showed that the algorithm generated values for epochs span the whole search space. Additionally, the level of exploration was higher than the one in run 1 (Fig. 15) and less than the one in run 2 (Fig. 16). Furthermore, the region between 60 and 80 was extensively exploited. Also, the algorithm then gets stuck in an epoch value of 32. Furthermore, as stated before the number of parameters held constant increase proportional to the number of iterations.

The other parameters also had the same phenomena occurring. In same cases the algorithm explored the entire search space, while in other cases it explored a few points then getting stuck in one of the values. Additionally, in other cases the algorithm gets to explore several values before it gets stuck on a single value.

8.6 Results Based on the Generation Accuracy

The accuracy values for each objective function calculation were observed and were plotted as shown in Fig. 18 respectively.

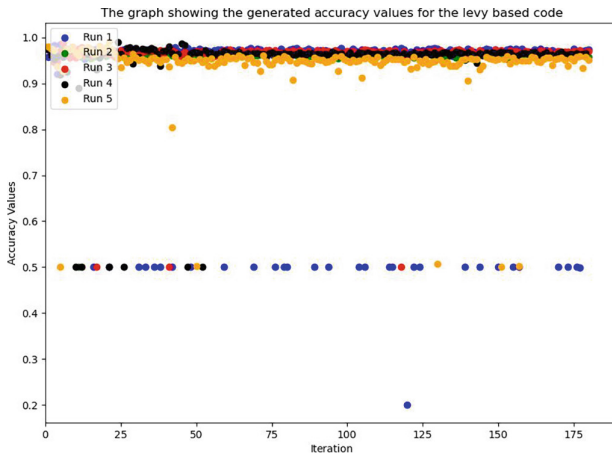


Fig. 18. Graph showing values for accuracy for the levy code

The graph in Fig. 18 shows a plot of the observed accuracy values. In addition, in Fig. 18 the bulk of the accuracy values (for all runs) were above 90%. Furthermore, there was no observable trend in the occurrence of the values that were 50% and below. Moreover, the number of these values were lowest in the Lévy based FA1→3 as compared to the other two implementations.

The proposed algorithm can easily be adopted for different problems that need parameter setting. This can be easily achieved by setting up the number of parameters and their respective upper and lower bounds. However, there are some identified limitations. Initially, the SMOTE augmentation strategy has several limitations that include computational expensiveness especially for large data sets. On the other hand, it would be ideal to train the proposed algorithm on different datasets to ensure variability in the training samples. Furthermore, the generated synthetic samples might not be a true representative of the true nature of the data.

9 Conclusion

The current study was based on the research carried out by Ghasemi et al. [20] (FA1→3 algorithm). Furthermore, the current research extended on the work by initially implementing the algorithm in Python, then suggesting an implementation alignment to the algorithm. Finally, the authors propose a Lévy based FA1→3 algorithm. Consequently, three experiments were conducted for each of the three versions of the algorithm. The first experiment had the highest recorded accuracy but the widest range of accuracy values. Additionally, the second experiment had a tighter range of accuracy values as compared to the first experiment. Finally, the last experiment had the tightest range of accuracy values. In addition, the algorithm's initial values for all the plots covered the search space and there was no observable trend in the generation of points. This was an improvement to the two other experiments that generated values that could be approximated by either a linear function or the equation shown in Eq. 25. In addition, a close look at the data generated indicated that the algorithm started holding one value constant then exploring other parameters while exploiting the value being held constant. In addition, the number of parameters held constant increased steadily. These indicate that the proposed algorithm both increased exploration and exploitation of the search space. Consequently, the performance of the algorithm was improved by enhancing the dependability of the algorithm. On the other hand, future work could consider, implementing the Lévy based FA1→3 in several other real life problems that include engineering problems. Moreover, the algorithm could be investigated and evaluated on its effectiveness on several bench marking functions using datasets such as CEC2021. In addition, researchers could consider a hybrid of the proposed algorithm with the leader and follower model. Also, the proposed algorithm could be used in conjunction with more effective data augmentation strategies. Furthermore, one could also consider implementing strategies to avoid an algorithm being stuck in a local minima.

References

1. Zojaji, Z., Atani, R.E., Monadjemi, A.H., et al.: A survey of credit card fraud detection techniques: Data and technique oriented perspective. arXiv preprint [arXiv:1611.06439](https://arxiv.org/abs/1611.06439) (2016)
2. F. B. of Investigation: Internet crime complaint center (ic3) (2023). Accessed 1 Mar 2024. <https://www.ic3.gov/>
3. Arun, G., Rajesh, P.: Design of metaheuristic feature selection with deep learning based credit card fraud detection model. In: 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), pp. 191–197. IEEE (2022)
4. Sam, U., Moses, G., Olajide, T.: Credit card fraud detection using machine learning algorithms (2023)
5. Dornadula, V.N., Geetha, S.: Credit card fraud detection using machine learning algorithms. *Procedia Comput. Sci.* **165**, 631–641 (2019)
6. Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., Bontempi, G.: Credit card fraud detection: a realistic modeling and a novel learning strategy. *IEEE Trans. Neural Networks Learn. Syst.* **29**(8), 3784–3797 (2017)
7. Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M., Anderla, A.: Credit card fraud detection-machine learning methods. In: 2019 18th International Symposium Infoteh-Jahorina (INFOTEH), pp. 1–5. IEEE (2019)
8. Priscilla, C.V., Prabha, D.P.: Influence of optimizing xgboost to handle class imbalance in credit card fraud detection. In: 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), pp. 1309–1315. IEEE (2020)
9. Novianady, T.R., Idroes, G.M., Maulana, A., Hardi, I., Ringga, E.S., Idroes, R.: Credit card fraud detection for contemporary financial management using XGBoost-driven machine learning and data augmentation techniques. *Indatu J. Manag. Account.* **1**(1), 29–35 (2023)
10. Ding, Y., Kang, W., Feng, J., Peng, B., Yang, A.: Credit card fraud detection based on improved variational autoencoder generative adversarial network. *IEEE Access* (2023)
11. Karthikeyan, T., Govindarajan, M., Vijayakumar, V.: An effective fraud detection using competitive swarm optimization based deep neural network. *Measur. Sens.* **27**, 100793 (2023)
12. Fiore, U., De Santis, A., Perla, F., Zanetti, P., Palmieri, F.: Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Inf. Sci.* **479**, 448–455 (2019)
13. Liu, X., et al.: NADS-RA: network anomaly detection scheme based on feature representation and data augmentation. *IEEE Access* **8**, pp. 214 781–214 800, 2020
14. Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., Bharath, A.A.: Generative adversarial networks: An overview. *IEEE Signal Process. Mag.* **35**(1), 53–65 (2018)
15. Asha, R., KR, S.K.: Credit card fraud detection using artificial neural network. *Global Trans. Proc.* **2**(1), 35–41 (2021)
16. Ali, A., et al.: Financial fraud detection based on machine learning: a systematic literature review. *Appl. Sci.* **12**(19), 9637 (2022)
17. Yang, X.-S.: Swarm intelligence based algorithms: a critical analysis. *Evol. Intel.* **7**, 17–28 (2014)
18. Padhi, B.K., Chakravarty, S., Naik, B., Pattanayak, R.M., Das, H.: RHSOFS: feature selection using the rock hyrax swarm optimization algorithm for credit card fraud detection system. *Sensors* **22**(23), 9321 (2022)

19. Zedadra, O., Guerrieri, A., Jouandeau, N., Spezzano, G., Seridi, H., Fortino, G.: Swarm intelligence-based algorithms within IoT-based systems: a review. *J. Parallel Distrib. Comput.* **122**, 173–187 (2018)
20. Ghasemi, M., Mohammadi, S.K., Zare, M., Mirjalili, S., Gil, M., Hemmati, R.: A new firefly algorithm with improved global exploration and convergence with application to engineering optimization. *Dec. Anal. J.* **5**, 100125 (2022)
21. Aydoğdu, İ., Akın, A., Saka, M.P.: Design optimization of real world steel space frames using artificial bee colony algorithm with levy flight distribution. *Adv. Eng. Softw.* **92**, 1–14 (2016)
22. Aydogdu, I.: Cost optimization of reinforced concrete cantilever retaining walls under seismic loading using a biogeography-based optimization algorithm with levy flights. *Eng. Optim.* **49**(3), 381–400 (2017)
23. Yıldız, B.S., Kumar, S., Pholdee, N., Bureerat, S., Sait, S.M., Yıldız, A.R.: A new chaotic levy flight distribution optimization algorithm for solving constrained engineering problems. *Expert. Syst.* **39**(8), e12992 (2022)
24. Zheng, T., et al.: Iwormlf: improved invasive weed optimization with random mutation and levy flight for beam pattern optimizations of linear and circular antenna arrays. *IEEE Access*, **8**, 19 460–19 478 (2020)
25. Sharma, H., Bansal, J.C., Arya, K., Yang, X.-S.: Lévy flight artificial bee colony algorithm. *Int. J. Syst. Sci.* **47**(11), 2652–2670 (2016)
26. Yıldız, B.S., Pholdee, N., Bureerat, S., Erdaş, M.U., Yıldız, A.R., Sait, S.M.: Comparison of the political optimization algorithm, the archimedes optimization algorithm and the levy flight algorithm for design optimization in industry. *Mater. Test.* **63**(4), 356–359 (2021)
27. Abualigah, L., Diabat, A., Elaziz, M.A.: Improved slime mould algorithm by opposition-based learning and levy flight distribution for global optimization and advances in real-world engineering problems. *J. Ambient Intell. Hum. Comput.* 1–40 (2023)
28. Liu, F., Sun, Y., Wang, G.-G., Wu, T.: An artificial bee colony algorithm based on dynamic penalty and levy flight for constrained optimization problems. *Arab. J. Sci. Eng.* **43**(12), 7189–7208 (2018)
29. Aydogdu, I., Carbas, S., Akin, A.: Effect of levy flight on the discrete optimum design of steel skeletal structures using metaheuristics. *Steel Compos. Struct.* **24**(1), 93–112 (2017)
30. Li, Y., Li, X., Liu, J., Ruan, X.: An improved bat algorithm based on levy flights and adjustment factors. *Symmetry* **11**(7), 925 (2019)
31. Vashishtha, G., Kumar, R.: An effective health indicator for the Pelton wheel using a levy flight mutated genetic algorithm. *Meas. Sci. Technol.* **32**(9), 094003 (2021)
32. Barisal, A.K., Panigrahi, T.K., Mishra, S.: A hybrid PSO-levy flight algorithm based fuzzy PID controller for automatic generation control of multi area power systems: fuzzy based hybrid PSO for automatic generation control. *Int. J. Energy Optim. Eng. (IJE OE)* **6**(2), 42–63 (2017)
33. Trivedi, I., Kumar, A., Ranpariya, A.H., Jangir, P.: Economic load dispatch problem with ramp rate limits and prohibited operating zones solve using levy flight moth-flame optimizer. In: 2016 International Conference on Energy Efficient Technologies for Sustainability (ICEETS), pp. 442–447. IEEE (2016)
34. Motamarri, R., Nagu, B.: GMPPT by using PSO based on levy flight for photovoltaic system under partial shading conditions. *IET Renew. Power Gener.* **14**(7), 1143–1155 (2020)

35. Motamarri, R., Bhookya, N.: Jaya algorithm based on lévy flight for global MPPT under partial shading in photovoltaic system. *IEEE J. Emerg. Sel. Top. Power Electron.* **9**(4), 4979–4991 (2020)
36. Duman, S., Kahraman, H.T., Guvenc, U., Aras, S.: Development of a lévy flight and FDB-based coyote optimization algorithm for global optimization and real-world acopf problems. *Soft. Comput.* **25**, 6577–6617 (2021)
37. Suja, K.R.: Mitigation of power quality issues in smart grid using levy flight based moth flame optimization algorithm. *J. Ambient. Intell. Humaniz. Comput.* **12**(10), 9209–9228 (2021). <https://doi.org/10.1007/s12652-020-02626-3>
38. Mokeddem, D.: Parameter extraction of solar photovoltaic models using enhanced levy flight based grasshopper optimization algorithm. *J. Electr. Eng. Technol.* **16**(1), 171–179 (2021)
39. Charin, C., Ishak, D., Zainuri, M., Ismail, B., Jamil, M.: A hybrid of bio-inspired algorithm based on levy flight and particle swarm optimizations for photovoltaic system under partial shading conditions. *Sol. Energy* **217**, 1–14 (2021)
40. Zhang, Y., Jin, Z., Zhao, X., Yang, Q.: Backtracking search algorithm with lévy flight for estimating parameters of photovoltaic models. *Energy Convers. Manage.* **208**, 112615 (2020)
41. Deotti, L.M.P., Pereira, J.L.R., da Silva Junior, I.C.: Parameter extraction of photovoltaic models using an enhanced lévy flight bat algorithm. *Energy Conv. Manag.* **221**, 113114 (2020)
42. Feng, Z.-K., Liu, S., Niu, W.-J., Li, S.-S., Wu, H.-J., Wang, J.-Y.: Ecological operation of cascade hydropower reservoirs by elite-guide gravitational search algorithm with lévy flight local search and mutation. *J. Hydrol.* **581**, 124425 (2020)
43. Zhang, Y., Chen, L., Li, Y., Zheng, X., Chen, J., Jin, J.: A hybrid approach for remaining useful life prediction of lithium-ion battery with adaptive levy flight optimized particle filter and long short-term memory network. *J. Energy Storage* **44**, 103245 (2021)
44. Mao, X., Song, S., Ding, F.: Optimal BP neural network algorithm for state of charge estimation of lithium-ion battery using PSO with levy flight. *J. Energy Storage* **49**, 104139 (2022)
45. Liu, M., Yao, X., Li, Y.: Hybrid whale optimization algorithm enhanced with lévy flight and differential evolution for job shop scheduling problems. *Appl. Soft Comput.* **87**, 105954 (2020)
46. Zhou, B., Liao, X.: Particle filter and levy flight-based decomposed multi-objective evolution hybridized particle swarm for flexible job shop greening scheduling with crane transportation. *Appl. Soft Comput.* **91**, 106217 (2020)
47. Tomassini, M., Antonioni, A.: Lévy flights and cooperation among mobile individuals. *J. Theor. Biol.* **364**, 154–161 (2015)
48. Zhou, T., Xu, S., Chen, X., Guo, A., Swingle, B.: Operator lévy flight: light cones in chaotic long-range interacting systems. *Phys. Rev. Lett.* **124**(18), 180601 (2020)
49. Cong, H., Zhi, L., Tian, Z., Zhu, A., Xu, C.: A multi-verse optimizer with levy flights for numerical optimization and its application in test scheduling for network-on-chip. *PLoS ONE* **11**(12), e0167341 (2016)
50. Barshandeh, S., Haghzadeh, M.: A new hybrid chaotic atom search optimization based on tree-seed algorithm and levy flight for solving optimization problems. *Eng. Comput.* **37**(4), 3079–3122 (2021)

51. Ekinci, S., Izci, D., Abu Zitar, R., Alsoud, A.R., Abualigah, L.: Development of lévy flight-based reptile search algorithm with local search ability for power systems engineering design problems. *Neural Comput. Appl.* **34**(22), 20 263–20 283 (2022)
52. Izci, D., Ekinci, S., Hekimoğlu, B.: Fractional-order PID controller design for buck converter system via hybrid lévy flight distribution and simulated annealing algorithm. *Arabian J. Sci. Eng.* **47**(11), 13 729–13 747 (2022)
53. Li, X., Niu, P., Liu, J.: Combustion optimization of a boiler based on the chaos and levy flight vortex search algorithm. *Appl. Math. Model.* **58**, 3–18 (2018)
54. Dou, R., Duan, H.: Lévy flight based pigeon-inspired optimization for control parameters optimization in automatic carrier landing system. *Aerosp. Sci. Technol.* **61**, 11–20 (2017)
55. Izci, D.: Design and application of an optimally tuned PID controller for dc motor speed regulation via a novel hybrid lévy flight distribution and nelder-mead algorithm. *Trans. Inst. Meas. Control.* **43**(14), 3195–3211 (2021)
56. Wang, W.-C., Xu, L., Chau, K.-W., Xu, D.-M.: Yin-yang firefly algorithm based on dimensionally Cauchy mutation. *Expert Syst. Appl.* **150**, 113216 (2020)
57. Yang, X.: Firefly algorithm, lévy distributions and global optimization (2010)
58. Farahani, S.M., Abshouri, A., Nasiri, B., Meybodi, M.: An improved firefly algorithm with directed movement. In: *Proceedings of 4th IEEE International Conference on Computer Science and Information Technology*, pp. 248–251 (2011)
59. Chou, J.-S., Ngo, N.-T.: Modified firefly algorithm for multidimensional optimization in structural design problems. *Struct. Multidiscip. Optim.* **55**, 2013–2028 (2017)
60. Tighzert, L., Fonlupt, C., Mendil, B.: A set of new compact firefly algorithms. *Swarm Evol. Comput.* **40**, 92–115 (2018)
61. Elhoseny, M.: Intelligent firefly-based algorithm with levy distribution (FF-l) for multicast routing in vehicular communications. *Expert Syst. Appl.* **140**, 112889 (2020)
62. Wu, J., Wang, Y.-G., Burrage, K., Tian, Y.-C., Lawson, B., Ding, Z.: An improved firefly algorithm for global continuous optimization problems. *Expert Syst. Appl.* **149**, 113340 (2020)
63. Alweshah, M., Abdullah, S.: Hybridizing firefly algorithms with a probabilistic neural network for solving classification problems. *Appl. Soft Comput.* **35**, 513–524 (2015)
64. Rajinikanth, V., Couceiro, M.: RGB histogram based color image segmentation using firefly algorithm. *Procedia Comput. Sci.* **46**, 1449–1457 (2015)
65. Kalantzis, G., Shang, C., Lei, Y., Leventouri, T.: Investigations of a GPU-based levy-firefly algorithm for constrained optimization of radiation therapy treatment planning. *Swarm Evol. Comput.* **26**, 191–201 (2016)
66. Kotteeswaran, R., Sivakumar, L.: Levy guided firefly algorithm based tuning of decentralised pi controller of nonlinear multivariable system-coal gasifier. *IFAC Proceedings Volumes* **47**(1), 127–134 (2014)
67. Yang, X.-S., Deb, S.: Eagle strategy using lévy walk and firefly algorithms for stochastic optimization. In: *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp. 101–111. Springer (2010)
68. Tan, S., Zhao, S., Wu, J.: QL-ADIFA: hybrid optimization using q-learning and an adaptive logarithmic spiral-levy firefly algorithm. *Math. Biosci. Eng.* **20**(8), 13 542–13 561 (2023)

69. Limas, A.V., Sudirwan, J., Fadlilah, S.N.: Perancangan sistem penjadwalan mesin hybrid flow shop dengan algoritma levyflight discrete firefly. *ComTech: Comput. Math. Eng. Appl.* 5(2), 671–684 (2014)
70. Fister, I., Yang, X.-S., Brest, J., Fister, I.: On the randomized firefly algorithm. *Cuckoo Search Firefly Algorithm Theory Appl.* 27–48 (2014)
71. Peng, Z., Dong, K., Yin, H., Bai, Y.: Modification of fish swarm algorithm based on levy flight and firefly behavior. *Comput. Intell. Neurosci.* (2018)
72. Singh, H., Kumar, A., Balyan, L.: A levy flight firefly optimizer based piecewise gamma corrected unsharp masking framework for satellite image enhancement. In: 2017 14th IEEE India Council International Conference (INDICON), pp. 1–5. IEEE (2017)
73. Hassanzadeh, T., Vojodi, H., Moghadam, A.M.E.: A multilevel thresholding approach based on levy-flight firefly algorithm. In: 2011 7th Iranian Conference on Machine Vision and Image Processing, pp. 1–5. IEEE (2011)
74. Liu, J., Shi, J., Hao, F., Dai, M., Zhang, Z.: A new firefly algorithm with enhanced attractiveness. In: 2021 5th International Conference on Automation, Control and Robots (ICACR), pp. 74–77. IEEE (2021)
75. Elangovan, D., Subedha, V.: An effective feature selection based classification model using firefly with levy and multilayer perceptron based sentiment analysis. In: 2020 International Conference on Inventive Computation Technologies (ICICT), pp. 376–380. IEEE (2020)
76. Zedadra, O., Guerrieri, A., Seridi, H.: LFA: a lévy walk and firefly-based search algorithm: application to multi-target search and multi-robot foraging. *Big Data Cogn. Comput.* **6**(1), 22 (2022)
77. Alghamdi, A.S.: Optimal power flow of renewable-integrated power systems using a gaussian bare-bones levy-flight firefly algorithm. *Front. Energy Res.* **10**, 921936 (2022)
78. Demri, M., Rahmoun, A., Omari, M.: Energy efficient clustering in wireless sensors network using adaptive lévy-flight firefly algorithm. *Int. J. Comput. Digit. Syst.* (2023)
79. Yong, X., Gao, Y., He, Y., Wang, H.: Improved firefly algorithm based on multi-strategy fusion. *J. Comput. Appl.* **42**(12), 3847 (2022)
80. Liu, G., et al.: Inversion of Rayleigh wave dispersion curve via adaptive logarithmic spiral-lévy firefly algorithm. *Chin. J. Geophys.* **66**(7), 3011–3025 (2023)
81. Sahoo, A., Chandra, S.: Lévy-flight firefly algorithm based active contour model for medical image segmentation. In: 2013 Sixth International Conference on Contemporary Computing (IC3), pp. 159–162. IEEE (2013)
82. Liu, S., Wang, Y.: A lévy flight based firefly algorithm for multilevel thresholding image segmentation. *J. Phys. Conf. Ser.* **1865**(4), 042098 (2021)
83. Shakarami, M.R., Faraji, I., Asghari, I., Akbari, M.: Optimal PID tuning for load frequency control using lévy-flight firefly algorithm. In: 2013 3rd International Conference on Electric Power and Energy Conversion Systems, pp. 1–5. IEEE (2013)
84. Peng, L., Zhang, D.: An adaptive lévy flight firefly algorithm for multilevel image thresholding based on rényi entropy. *J. Supercomput.* 1–22 (2022)
85. Elangovan, D., Subedha, V.: Automatic sentimental analysis by firefly with levy and multilayer perceptron. *Comput. Syst. Sci. Eng.* **46**(3) (2023)
86. Hussain, M., Jenkins, W., Radhakrishnan, C.: Use of the lévy flight firefly algorithm in 2D MCCLELLAN unconstrained transform adaptive filters. In: 2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWS-CAS), pp. 856–859. IEEE (2020)

87. Zhang, X., Wang, S.: Firefly search algorithm based on leader strategy. *Eng. Appl. Artif. Intell.* **123**, 106328 (2023)
88. Lv, L., Zhao, J.: The firefly algorithm with gaussian disturbance and local search. *J. Signal Process. Syst.* **90**, 1123–1131 (2018)
89. Gandomi, A.H., Yang, X.-S., Talatahari, S., Alavi, A.H.: Firefly algorithm with chaos. *Commun. Nonlinear Sci. Numer. Simul.* **18**(1), 89–98 (2013)
90. Dhal, K.G., Quraishi, I., Das, S., et al.: A chaotic lévy flight approach in bat and firefly algorithm for gray level image enhancement. *Int. J. Image Graphics Signal Process. (IJIGSP)* **7**(7), 69 (2015)
91. Dhal, K.G., Iqbal Quraishi, M., Das, S.: Development of firefly algorithm via chaotic sequence and population diversity to enhance the image contrast. *Nat. Comput.* **15**, 307–318 (2016)
92. Hassanzadeh, T., Kanan, H.R.: Fuzzy fa: a modified firefly algorithm. *Appl. Artif. Intell.* **28**(1), 47–65 (2014)
93. Wang, H., Wang, W., Sun, H., Rahnamayan, S.: Firefly algorithm with random attraction. *Int. J. Bio-Inspired Comput.* **8**(1), 33–41 (2016)
94. Sarbazfard, S., Jafarian, A.: A hybrid algorithm based on firefly algorithm and differential evolution for global optimization. *Int. J. Adv. Comput. Sci. Appl.* **7**(6) (2016)
95. Nokhwal, S., Chandrasekharan, M., Chaudhary, A.: Secure information embedding in images with hybrid firefly algorithm. *arXiv preprint [arXiv:2312.13519](https://arxiv.org/abs/2312.13519)* (2023)
96. Zare, M., et al.: A global best-guided firefly algorithm for engineering problems. *J. Bionic Eng.* **20**(5), 2359–2388 (2023)
97. Ritthipakdee, A., Thammano, A., Premasathian, N., Uyyanonvara, B.: An improved firefly algorithm for optimization problems. In: *The 5th International Symposium on Advanced Control of Industrial Processes (ADCONIP 2014)*, pp. 159–164 (2014)
98. Ren, H., Ren, H., Sun, Z.: HSFA: a novel firefly algorithm based on a hierarchical strategy. *Knowl.-Based Syst.* **279**, 110950 (2023)
99. Makki, S., Assaghir, Z., Taher, Y., Haque, R., Hacid, M.-S., Zeineddine, H.: An experimental study with imbalanced classification approaches for credit card fraud detection. *IEEE Access* **7**, 93 010–93 022 (2019)
100. Sadgali, I., Sael, N., Benabbou, F.: Detection of credit card fraud: state of art. *Int. J. Comput. Sci. Netw. Secur.* **18**(11), 76–83 (2018)
101. Georgieva, S., Markova, M., Pavlov, V.: Using neural network for credit card fraud detection. In: *AIP Conference Proceedings*, vol. 2159, no. 1. AIP Publishing (2019)
102. Tiwari, P., Mehta, S., Sakhuja, N., Kumar, J., Singh, A.K.: Credit card fraud detection using machine learning: a study. *arXiv preprint [arXiv:2108.10005](https://arxiv.org/abs/2108.10005)* (2021). <https://doi.org/10.48550/arXiv.2012.03754>
103. Gao, Y., Zhang, S., Lu, J.: Machine learning for credit card fraud detection. In: *Proceedings of the 2021 1st International Conference on Control and Intelligent Robotics*, pp. 213–219 (2021). <https://doi.org/10.1145/3473714.3473749>
104. Trivedi, N.K., Simaiya, S., Lilhore, U.K., Sharma, S.K.: An efficient credit card fraud detection model based on machine learning methods. *Int. J. Adv. Sci. Technol.* **29**(5), 3414–3424 (2020)
105. Fang, Y., Zhang, Y., Huang, C.: Credit card fraud detection based on machine learning. *Comput. Mat. Continua* **61**(1) (2019)
106. Sailusha, R., Gnaneswar, V., Ramesh, R., Rao, G.R.: Credit card fraud detection using machine learning. In: *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1264–1270. IEEE (2020)

107. Alfaiz, N.S., Fati, S.M.: Enhanced credit card fraud detection model using machine learning. *Electronics* **11**(4), 662 (2022)
108. Khatri, S., Arora, A., Agrawal, A.P.: Supervised machine learning algorithms for credit card fraud detection: a comparison. In: 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), pp. 680–683. IEEE (2020)
109. Suryanarayana, S.V., Balaji, G., Rao, G.V., et al.: Machine learning approaches for credit card fraud detection. *Int. J. Eng. Technol* **7**(2), 917–920 (2018)
110. Dal Pozzolo, A.: Adaptive machine learning for credit card fraud detection (2015)
111. Save, P., Tiwarekar, P., Jain, K.N., Mahyavanshi, N.: A novel idea for credit card fraud detection using decision tree. *Int. J. Comput. Appl.* **161**(13) (2017)
112. de Sá, A.G., Pereira, A.C., Pappa, G.L.: A customized classification algorithm for credit card fraud detection. *Eng. Appl. Artif. Intell.* **72**, 21–29 (2018)
113. Taha, A.A., Malebary, S.J.: An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE Access* **8**, 25 579–25 587 (2020)
114. Kho, J.R.D., Vea, L.A.: Credit card fraud detection based on transaction behavior. In: TENCON 2017-2017 IEEE Region 10 Conference, pp. 1880–884. IEEE (2017)
115. Kumar, M.S., Soundarya, V., Kavitha, S., Keerthika, E., Aswini, E.: Credit card fraud detection using random forest algorithm. In: 2019 3rd International Conference on Computing and Communications Technologies (ICCCT), pp. 149–153. IEEE (2019)
116. Maniraj, S., Saini, A., Ahmed, S., Sarkar, S.: Credit card fraud detection using machine learning and data science. *Int. J. Eng. Res.* **8**(9), 110–115 (2019)
117. Xuan, S., Liu, G., Li, Z., Zheng, L., Wang, S., Jiang, C.: Random forest for credit card fraud detection. In: 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), pp. 1–6. IEEE (2018)
118. Porwal, U., Mukund, S.: Credit card fraud detection in e-commerce. In: 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), pp. 280–287. IEEE (2019)
119. Behera, T.K., Panigrahi, S.: Credit card fraud detection: a hybrid approach using fuzzy clustering & neural network. In: 2015 Second International Conference on Advances in Computing and Communication Engineering, pp. 494–499. IEEE (2015)
120. Tran, P.H., Tran, K.P., Huong, T.T., Heuchenne, C., HienTran, P., Le, T.M.H.: Real time data-driven approaches for credit card fraud detection. In: Proceedings of the 2018 International Conference on E-Business and Applications, pp. 6–9 (2018). <https://doi.org/10.1145/3194188.3194196>
121. Banerjee, R., Bourla, G., Chen, S., Kashyap, M., Purohit, S.: Comparative analysis of machine learning algorithms through credit card fraud detection. In: 2018 IEEE MIT Undergraduate Research Technology Conference (URTC), pp. 1–4. IEEE (2018)
122. Chen, J.I.-Z., Lai, K.-L.: Deep convolution neural network model for credit-card fraud detection and alert. *J. Artif. Intell.* **3**(02), 101–112 (2021)
123. Awoyemi, J.O., Adetunmbi, A.O., Oluwadare, S.A.: Credit card fraud detection using machine learning techniques: a comparative analysis. In: 2017 International Conference on Computing Networking and Informatics (ICCNI), pp. 1–9. IEEE (2017)

124. Zhang, X., Han, Y., Xu, W., Wang, Q.: HOBA: a novel feature engineering methodology for credit card fraud detection with a deep learning architecture. *Inf. Sci.* **557**, 302–316 (2021)
125. O. Voican, “Credit card fraud detection using deep learning techniques.” *Informatica Economica*, vol. 25, no. 1, 2021
126. Nguyen, T.T., Tahir, H., Abdelrazek, M., Babar, A.: Deep learning methods for credit card fraud detection. arXiv preprint [arXiv:2012.03754](https://arxiv.org/abs/2012.03754) (2020). <https://doi.org/10.48550/arXiv.2012.03754>
127. Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., Beling, P.: Deep learning detecting fraud in credit card transactions. In: 2018 Systems and Information Engineering Design Symposium (SIEDS), pp. 129–134. IEEE (2018)
128. Lebichot, B., Le Borgne, Y.-A., He-Guelton, L., Oblé, F., Bontempi, G.: Deep-learning domain adaptation techniques for credit cards fraud detection. In: Recent Advances in Big Data and Deep Learning: Proceedings of the INNS Big Data and Deep Learning Conference INNSBDDL2019, held at Sestri Levante, Genova, Italy 16–18 April 2019, pp. 78–88. Springer (2020)
129. Pillai, T.R., Hashem, I.A.T., Brohi, S.N., Kaur, S., Marjani, M.: Credit card fraud detection using deep learning technique. In: 2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA), pp. 1–6. IEEE (2018)
130. Zeager, M.F., Sridhar, A., Fogal, N., Adams, S., Brown, D.E., Beling, P.A.: Adversarial learning in credit card fraud detection. In: 2017 Systems and Information Engineering Design Symposium (SIEDS), pp. 112–116. IEEE (2017)
131. Jurgovsky, J., et al.: Sequence classification for credit-card fraud detection. *Expert Syst. Appl.* **100**, 234–245 (2018)
132. Carcillo, F., Le Borgne, Y.-A., Caelen, O., Kessaci, Y., Oblé, F., Bontempi, G.: Combining unsupervised and supervised learning in credit card fraud detection. *Inf. Sci.* **557**, 317–331 (2021)
133. Esenogho, E., Mienye, I.D., Swart, T.G., Aruleba, K., Obaido, G.: A neural network ensemble with feature engineering for improved credit card fraud detection. *IEEE Access* **10**, 16 400–16 407 (2022)
134. Forough, J., Momtazi, S.: Ensemble of deep sequential models for credit card fraud detection. *Appl. Soft Comput.* **99**, 106883 (2021)
135. Bagga, S., Goyal, A., Gupta, N., Goyal, A.: Credit card fraud detection using pipeling and ensemble learning. *Procedia Comput. Sci.* **173**, 104–112 (2020)
136. Sohony, I., Pratap, R., Nambiar, U.: Ensemble learning for credit card fraud detection. In: Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, pp. 289–294 (2018). <https://doi.org/10.1145/3152494.3156815>
137. Karthik, V., Mishra, A., Reddy, U.S.: Credit card fraud detection by modelling behaviour pattern using hybrid ensemble model. *Arab. J. Sci. Eng.* **47**(2), 1987–1997 (2022)
138. Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., Bontempi, G.: Credit card fraud detection and concept-drift adaptation with delayed supervised information. In: 2015 International Joint Conference on Neural networks (IJCNN), pp. 1–8. IEEE (2015)
139. Carcillo, F., Dal Pozzolo, A., Le Borgne, Y.-A., Caelen, O., Mazzer, Y., Bontempi, G.: Scarff: a scalable framework for streaming credit card fraud detection with spark. *Inf. Fusion* **41**, 182–194 (2018)

140. Lebichot, B., Verhelst, T. Le Borgne, Y.-A., He-Guelton, L., Oble, F., Bontempi, G.: Transfer learning strategies for credit card fraud detection. *IEEE Access* **9**, 114 754–114 766 (2021)
141. Patil, S., Nemade, V., Soni, P.K.: Predictive modelling for credit card fraud detection using data analytics. *Procedia Comput. Sci.* **132**, 385–395 (2018)
142. Vengatesan, K., Kumar, A., Yuvraj, S., Kumar, V., Sabnis, S.: Credit card fraud detection using data analytic techniques. *Adv. Math. Sci. J.* **9**(3), 1185–1196 (2020)
143. Carneiro, N., Figueira, G., Costa, M.: A data mining based system for credit-card fraud detection in e-tail. *Decis. Support Syst.* **95**, 91–101 (2017)
144. Porwal, U., Mukund, S.: Credit card fraud detection in e-commerce: an outlier detection approach. *arXiv preprint [arXiv:1811.02196](https://arxiv.org/abs/1811.02196)* (2018). <https://doi.org/10.48550/arXiv.1811.02196>
145. Yee, O.S., Sagadevan, S., Malim, N.H.A.H.: Credit card fraud detection using machine learning as data mining technique. *J. Telecommun. Electronic Comput. Eng. (JTEC)* **10**(1–4), 23–27 (2018)
146. Suresh, G., Raj, R.J.: A study on credit card fraud detection using data mining techniques. *Int. J. Data Min. Techniques Appl.* **7**(1), 21–24 (2018)
147. Malini, N., Pushpa, M.: Analysis on credit card fraud identification techniques based on KNN and outlier detection. In: 2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-informatics (AEEICB), pp. 255–258. *IEEE* (2017)
148. Randhawa, K., Loo, C.K., Seera, M., Lim, C.P., Nandi, A.K.: Credit card fraud detection using adaboost and majority voting. *IEEE Access*, vol. 6, pp. 14 277–14 284 (2018)
149. Wang, D., Chen, B., Chen, J.: Credit card fraud detection strategies with consumer incentives. *Omega* **88**, 179–195 (2019)
150. Jiang, C., Song, J., Liu, G., Zheng, L., Luan, W.: Credit card fraud detection: a novel approach using aggregation strategy and feedback mechanism. *IEEE Internet Things J.* **5**(5), 3637–3647 (2018)
151. Mavrovouniotis, M., Li, C., Yang, S.: A survey of swarm intelligence for dynamic optimization: algorithms and applications. *Swarm Evol. Comput.* **33**, 1–17 (2017)
152. Alam, S., Dobbie, G., Koh, Y.S., Riddle, P., Rehman, S.U.: Research on particle swarm optimization based clustering: a systematic review of literature and techniques. *Swarm Evol. Comput.* **17**, 1–13 (2014)
153. Fister, I., Fister, I., Jr., Yang, X.-S., Brest, J.: A comprehensive review of firefly algorithms. *Swarm Evol. Comput.* **13**, 34–46 (2013)
154. Kumar, V., Kumar, D.: A systematic review on firefly algorithm: past, present, and future. *Archives Comput. Meth. Eng.* **28**, 3269–3291 (2021)
155. Chakraborty, A., Kar, A.K.: Swarm intelligence: a review of algorithms. In: *Nature-Inspired Computing and Optimization: Theory and Applications*, pp. 475–494 (2017)
156. Gupta, A.K., Verma, A.K., Sadawarti, H.: Analysis of various swarm-based & ant-based algorithms. In: *Proceedings of the International Conference on Advances in Computing and Artificial Intelligence*, pp. 39–43 (2011)
157. Tan, Y., Ding, K.: A survey on GPU-based implementation of swarm intelligence algorithms. *IEEE Trans. Cybernet.* **46**(9), 2028–2041 (2015)
158. Hussain, K., Salleh, M., Cheng, S., Shi, Y.: On the exploration and exploitation in popular swarm-based metaheuristic algorithms. *Neural Comput. Appl.* **31**(11), 7665–7683 (2019)

159. Farahani, S.M., Abshouri, A.A., Nasiri, B., Meybodi, M.: A gaussian firefly algorithm. *Int. J. Mach. Learn. Comput.* **1**(5), 448 (2011)
160. Ali, N., Othman, M.A., Husain, M.N., Misran, M.H.: A review of firefly algorithm. *ARPN J. Eng. Appl. Sci.* **9**(10), 1732–1736 (2014)
161. Senthilnath, J., Omkar, S., Mani, V.: Clustering using firefly algorithm: performance study. *Swarm Evol. Comput.* **1**(3), 164–171 (2011)
162. Xie, H., et al.: Improving k-means clustering with enhanced firefly algorithms. *Appl. Soft Comput.* **84**, 105763 (2019)
163. Kazemzadeh-Parsi, M.-J.: A modified firefly algorithm for engineering design optimization problems. *Iranian J. Sci. Technol. Trans. Mech. Eng.* **38**(M2), 403 (2014)
164. Tilahun, S.L., Ngnotchouye, J., Hamadneh, N.N.: Continuous versions of firefly algorithm: a review. *Artif. Intell. Rev.* **51**, 445–492 (2019)
165. Subramanian, R., Thanushkodi, K.: An efficient firefly algorithm to solve economic dispatch problems. *Int. J. Soft Comput. Eng.* **2**(1), 52–55 (2013)
166. Liu, C., Zhao, Y., Gao, F., Liu, L., et al.: Three-dimensional path planning method for autonomous underwater vehicle based on modified firefly algorithm. *Math. Probl. Eng.* **2015** (2015)
167. dos Santos Coelho, L., Mariani, V.C.: Firefly algorithm approach based on chaotic tinkerbelle map applied to multivariable PID controller tuning. *Comput. Math. Appl.* **64**(8), 2371–2382 (2012)
168. Yan, X., Zhu, Y., Wu, J., Chen, H.: An improved firefly algorithm with adaptive strategies. *Adv. Sci. Lett.* **16**(1), 249–254 (2012)
169. Selvarasu, R., Kalavathi, M.S., Rajan, C.C.A.: SVC placement for voltage constrained loss minimization using self-adaptive firefly algorithm. *Arch. Electr. Eng.* **62**(4) (2013)
170. Barthelemy, P., Bertolotti, J., Wiersma, D.S.: A lévy flight for light. *Nature* **453**(7194), 495–498 (2008)
171. Dubkov, A.A., Spagnolo, B., Uchaikin, V.V.: Lévy flight superdiffusion: an introduction. *Int. J. Bifurcation Chaos* **18**(09), 2649–2672 (2008)
172. Barndorff-Nielsen, O.E., Mikosch, T., Resnick, S.I.: Lévy Processes: Theory and Applications. Springer (2001)
173. Viswanathan, G.M., et al.: Lévy flights in random searches. *XXPhys. A* **282**(1–2), 1–12 (2000)
174. Bacanin, N.: An object-oriented software implementation of a novel cuckoo search algorithm. In: *Proceedings of the 5th European Conference on European Computing Conference (ECC'11)*, pp. 245–250 (2011)
175. Jensi, R., Jiji, G.W.: An enhanced particle swarm optimization with levy flight for global optimization. *Appl. Soft Comput.* **43**, 248–261 (2016)
176. Hakh, H., Uğuz, H.: A novel particle swarm optimization algorithm with levy flight. *Appl. Soft Comput.* **23**, 333–345 (2014)
177. Xie, J., Zhou, Y., Chen, H.: A novel bat algorithm based on differential operator and lévy flights trajectory. In: *Computational Intelligence and Neuroscience*, vol. 2013 (2013)
178. Chechkin, A., Metzler, R., Klafter, J., Yu Gonchar, V.: Introduction to the theory of levy flights. anomalous transport: Foundations and applications. *Anomalous Transport: Foundations and Applications*, pp. 129–162. Weinheim, Germany: John Wiley & Sons, Inc (2008)
179. Chawla, M., Duhan, M.: Levy flights in metaheuristics optimization algorithms-a review. *Appl. Artif. Intell.* **32**(9–10), 802–821 (2018)
180. Mantegna, R.N.: Fast, accurate algorithm for numerical simulation of levy stable stochastic processes. *Phys. Rev. E* **49**(5), 4677 (1994)

181. Emary, E., Zawbaa, H.M., Sharawi, M.: Impact of lèvy flight on modern meta-heuristic optimizers. *Appl. Soft Comput.* **75**, 775–789 (2019)
182. Ling, Y., Zhou, Y., Luo, Q.: Lévy flight trajectory-based whale optimization algorithm for global optimization. *IEEE Access* **5**, 6168–6186 (2017)
183. Dalpozz, A.A.: Kaggle: credit card fraud detection 202. Accessed 1 Mar 2024. <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>



Juro: A Retrieval-Augmented Generation AI Chatbot for Enhancing Legal Information Access in Resource-Constrained Settings

Bernard Ngandu¹(✉) , Landry Mbale¹ , and Antoine Bagula^{1,2}

¹ Université Nouveaux Horizons, Lubumbashi, Democratic Republic of the Congo
ngandubernard@gmail.com, abagula@uwc.ac.za

² University of the Western, Cape, South Africa

Abstract. Access to legal information remains a significant challenge in resource-constrained settings where the digitization of legal systems is still in its early stages. To address this issue, we developed **Juro**, an AI-based chatbot architecture utilizing a Retrieval-Augmented Generation (RAG) framework. Leveraging a curated dataset of over 8,400 legal documents, **Juro** provides a user-friendly platform that simplifies complex legal language and ensures information reliability through a robust source citation mechanism. This paper demonstrates the applicability of adaptable AI-driven solutions in low-resource environments, offering a flexible chatbot architecture that can be tailored to various contexts where information accessibility remains a critical challenge. The legal system in the Democratic Republic of Congo (DRC) serves as a use case, illustrating the potential of **Juro** in addressing similar challenges across developing countries.

Keywords: Artificial Intelligence · Chatbot · Information Accessibility · Large Language Model · Legal System · Retrieval-Augmented Generation

1 Introduction

Access to legal information is a cornerstone of justice and the rule of law, yet it remains a significant challenge in resource-constrained settings, particularly in developing regions where digital infrastructure is limited. These regions often have complex legal systems characterized by an interplay of formal legal norms and local customs. This complexity, coupled with a severe scarcity of digital resources, poses significant barriers for citizens trying to navigate and comprehend the legal landscape. The digitization of legal systems in these settings is often in its infancy, which further exacerbates the difficulty in accessing reliable legal information.

Recent advancements in Natural Language Processing (NLP), especially with the advent of Large Language Models (LLMs) [5, 15], have shown considerable promise in making legal information more accessible by simplifying complex legal texts. However, applying LLMs in resource-constrained settings presents unique challenges. The effectiveness of these models is often limited by their generalization ability in specific domains, and their use without sufficient context can lead to inaccuracies (hallucinations), which can have serious consequences in legal contexts.

This paper introduces **Juro**, an AI-based chatbot architecture utilizing a Retrieval-Augmented Generation (RAG) framework [11]. Designed as a flexible, model-agnostic system, Juro supports interchangeable LLMs and embeddings models. In this work, we apply Juro within the legal context of the DRC as a primary use case, demonstrating its potential to improve access to information in resource-constrained environments. This flexibility allows for the integration of either open-source models, which support data privacy in professional applications, or proprietary models, which enhance the accessibility of knowledge for the general public. While Juro is applied here with a focus on the unique needs and challenges of the DRC, its adaptable design makes it broadly applicable to other developing countries facing similar constraints, offering a scalable solution for improving access to information across a range of resource-constrained environments.

While LLMs are increasingly recognized as powerful tools worldwide, significant research gaps remain, particularly in their application to specialized domains like the legal sector in developing countries. This paper focuses on two critical issues in the deployment of LLMs: effective context integration within the RAG framework and domain-specific adaptation of foundation models. Although challenges such as overfitting and transferability are acknowledged, they are beyond the scope of this initial work.

The contributions of this research are threefold:

- (a) We introduce **Juro**, an AI-based chatbot architecture leveraging a RAG framework, specifically designed for resource-constrained settings, with the legal system in the DRC serving as a use case to address the unique challenges posed by such environments.
- (b) We develop a comprehensive legal corpus, drawn from over 8,400 legal documents, which serves as the foundational knowledge base for **Juro**.
- (c) We demonstrate through empirical evaluation that **Juro** outperforms existing chatbots and models in legal question-answering tasks within the context of the DRC, thereby validating the efficacy of our approach and its potential for broader application in other developing countries.

The remainder of this paper is organized as follows:

- Section 2 reviews the related work in the field of legal chatbots, RAG, and LLMs.
- Section 3 outlines the architecture of Juro, providing a detailed overview of its components and explaining how the RAG framework is integrated.

- Section 4 presents the methodology used to design and develop Juro.
- Section 5 outlines the experimental settings and process used to evaluate Juro
- Section 6 discusses the results obtained from the empirical evaluation of Juro in legal question-answering tasks.
- Finally, Sect. 7 concludes the paper with insights into future work, including scaling the system and improving domain-specific adaptations for enhanced accuracy in different contexts.

2 Related Works

In developing **Juro**, it is essential to contextualize our work within the broader landscape of legal technologies, particularly those involving LLMs and Legal Benchmarks. The focus on these two areas is deliberate, as they are critical in understanding the current capabilities and limitations of AI-driven legal solutions and in identifying the gaps that our work aims to address. Legal LLMs represent the cutting-edge in applying AI to the legal domain, while Legal Benchmarks provide the necessary frameworks to evaluate these models' performance in a rigorous and standardized manner.

2.1 Legal Large Language Models

Legal LLMs have become a focal point in legal technology due to their ability to process and generate complex legal text, a task that requires deep understanding and contextual awareness. These models are typically fine-tuned from pre-trained language models or trained from scratch using large datasets of legal documents, statutes, case law, and other legal texts. This specialization enables them to tackle various legal tasks with greater accuracy and relevance compared to general-purpose LLMs. Our interest in Legal LLMs stems from their potential to bridge the accessibility gap in legal information, particularly in resource-constrained settings where legal expertise and resources are limited. Inspired by these models, our work aims to enhance the accessibility and comprehensibility of legal information for the general public and professionals alike. Earlier efforts in this domain focused on specific legal tasks, such as legal judgment prediction, document and case retrieval, and question answering. For instance, Chalkidis et al. [1] explored legal judgment prediction, demonstrating how hierarchical models can outperform traditional methods in predicting court case outcomes. Luo et al. [12] and Kien et al. [9] advanced the field of legal document retrieval and question answering by integrating domain-specific knowledge into their models, improving accuracy and task-specific performance. Recent developments have seen the emergence of specialized legal LLMs, such as **LawGPT** [20], **SaulLM-7B** [2], **LawGPT1.0** [14], and **DISC-LawLLM** [19], which are fine-tuned on vast legal corpora. These models are tailored to execute complex legal tasks, making them more effective tools for legal practitioners. However, one significant challenge in their application is the hallucination problem, where models generate information that is not grounded in the source data. To mitigate this, some models, such

as **ChatLaw**, **Wisdom-Interrogatory**, and **LawyerLLaMA** [8, 10], incorporate legal data retrieval mechanisms, ensuring that outputs are accurate and reliable. Our work builds on these advancements. By focusing on Legal LLMs, we position our research at the intersection of cutting-edge AI and real-world legal challenges in developing countries.

2.2 Legal Benchmarks

The evaluation of LLMs in legal contexts requires robust benchmarks that can comprehensively assess their performance across various tasks. Legal Benchmarks are essential tools for this purpose, as they provide standardized metrics and tasks that allow researchers to measure and compare the effectiveness of different models. Our focus on Legal Benchmarks is driven by the need to ensure that **Juro**, not only performs well in controlled settings but also addresses real-world legal information needs in resource-constrained environments. By benchmarking our system against existing models, we can validate its efficacy and identify areas for improvement. **LawBench** is one such comprehensive benchmark that includes 20 legal tasks categorized into three cognitive levels: memorization, understanding, and application [6]. This benchmark challenges LLMs to not only recall legal information but also apply legal principles to new scenarios, thus testing their ability to reason like a legal expert. **LAiW** is another benchmark that evaluates models across three levels: basic information retrieval, legal foundation inference, and complex legal application [3]. These benchmarks are crucial for assessing a model's ability to infer legal rules from texts and apply them to specific problems, which is vital for tasks such as legal drafting and advising. **SimuCourt**, a more specialized benchmark, introduces a judicial decision-making task, assessing a model's capability to analyze legal cases and render informed judgments [7]. This evaluation simulates the role of a judge and tests the model's understanding of legal nuances and its ability to apply legal reasoning.

2.3 Addressing the Gaps

While these benchmarks are invaluable for assessing the generative capabilities of LLMs in the legal domain, they often overlook the importance of the retrieval component, especially in the context of RAG systems. This gap is particularly relevant to our work, as the effectiveness of a RAG system in legal contexts depends heavily on its ability to retrieve accurate and relevant information from a vast corpus of legal texts. To address this, **LegalBench-RAG** was developed as the first benchmark specifically designed to evaluate the retrieval step within RAG pipelines in legal contexts [17]. This benchmark emphasizes precise retrieval, focusing on extracting minimal, highly relevant text segments from legal documents. Precise retrieval is crucial because it prevents issues such as exceeding context window limitations in LLMs, which can lead to increased processing costs, latency, and potential errors in outputs. Furthermore, accurate retrieval enables LLMs to generate reliable citations, maintaining the credibility

of the information provided. Inspired by these benchmarks for our evaluation process, we ensure that **Juro** meets the highest standards of performance and reliability, making it a valuable tool for enhancing legal information accessibility in resource-constrained settings.

3 Juro Architecture Description

The architecture presented in Fig. 1 is based on a **Retrieval-Augmented Generation (RAG)** framework. This architecture enhances the capabilities of a LLM by incorporating external knowledge sources, improving information retrieval and response generation. Below is a description of the key building blocks of the Juro architecture:

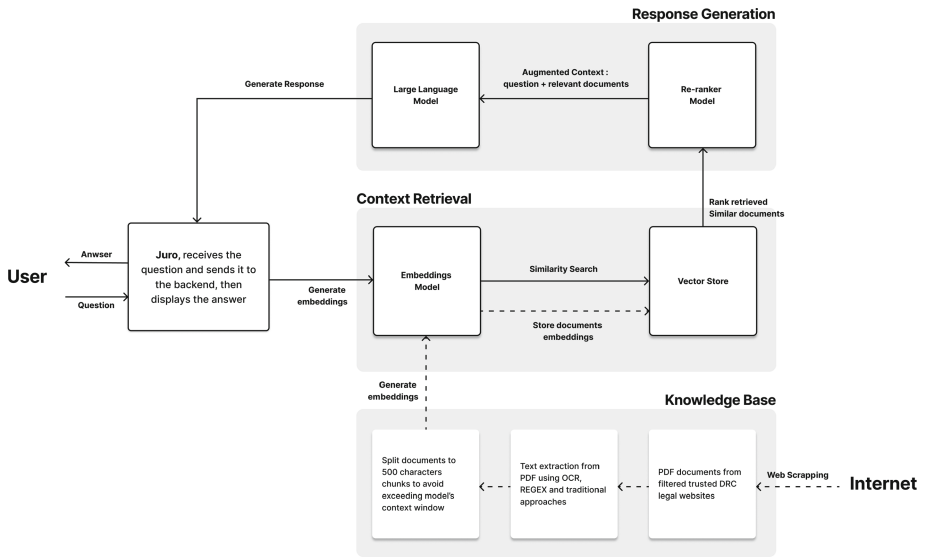


Fig. 1. Juro's architecture

3.1 Frontend

- **User Input:** The user interacts with the system through the frontend, submitting questions.
- **Query Processing:** The frontend receives the user's input and sends it to the backend for processing. Basic cross-checking might be performed here to ensure the query is properly structured.
- **Answer Generation:** After the backend processes the query and retrieves relevant data, the generated response is sent back to the frontend to be displayed to the user.

3.2 Backend

The backend forms the core of architecture, consisting of several components for document retrieval, embeddings generation, and LLM processing.

- **Symphony Web Framework:** This is the orchestration layer that connects the various backend components¹. It routes the user's query and manages data flow between different services in the backend².
- **Document Storage:**
 - **Raw Documents (Web Crawled):** These are documents collected from external sources (e.g., through web crawling) that form the knowledge base used in our architecture.
 - **Split Documents:** Raw documents are pre-processed and split into smaller, manageable chunks for efficient retrieval and processing.
- **Embedding Model:** The embedding model generates vector representations (embeddings) of the text in the documents. These embeddings capture the semantic meaning of the text, which is essential for performing semantic searches rather than simple keyword-based searches.
- **Vector Store:** Once embeddings are generated, they are stored in the vector store. The vector store enables efficient similarity searches based on the vector representations of the documents. It helps to retrieve relevant documents that are semantically similar to the user's query.
- **Large Language Model (LLM):** The LLM generates the final answer by integrating the retrieved document chunks into the response. It utilizes the retrieved documents to ensure that the generated response is accurate and contextually relevant.

3.3 Question-Answering Process Flow

The interaction between the different components is as follows (see Fig. 1):

1. The user submits a query through the frontend.
2. The query is sent to the backend, where it is processed by an embedding model to generate a vector representation of the query.
3. A similarity search is performed using the query's vector representation to retrieve the top relevant document chunks from the vector store.
4. The retrieved document chunks are passed through a re-ranker model, which prioritizes and ranks them by relevance to ensure the most pertinent information is considered.
5. The LLM takes the top-ranked document chunks and uses them to generate an accurate and contextually relevant answer to the user's query.
6. The generated answer is returned to the frontend, where it is displayed to the user.

¹ <https://github.com/theodo-group/LLPhant>.

² <https://symfony.com/>.

4 Methodology

Our methodology is structured around a layered architecture, specifically focusing on **Knowledge Base, Context Retrieval, and Response Generation**. Each layer is designed to address different aspects of processing and generating legal knowledge pertinent to the DRC.

4.1 Knowledge Base Layer

The knowledge base of our architecture involves the creation of a comprehensive and contextually relevant dataset. This dataset is crucial for generating accurate and contextually relevant response. The process of dataset creation includes several key steps:

1. **Document Discovery:** To address the scarcity of official digital legal resources in the DRC, we employed the Google Custom Search API³. We used a set of carefully selected keywords related to Congolese law to generate a broad list of URLs containing legal information.
2. **Filtering and Classification:** To refine the dataset, we utilized the nli-MiniLM2-L6-H768 as a zero-shot classification model⁴. This model helped classify URLs based on their relevance to Congolese legal context. The filtered URLs were then manually verified by local legal experts to ensure the dataset comprised authoritative and legit legal documents.
3. **Document Collection:** We employed web scraping techniques to collect 8,408 PDF documents from the filtered URLs (see Table 1). The documents were processed using Optical Character Recognition (OCR) with the Tesseract OCR Engine [18], regular expressions (REGEX), and standard text extraction techniques to handle diverse formats and layouts.

The dataset creation process is essential for building a solid foundation for subsequent layers, ensuring that the models used in those layers have access to relevant and high-quality legal information.

4.2 Context Retrieval Layer

This layer focuses on adapting general-purpose LLM to the specific legal domain by generating and storing embeddings. This layer is crucial for enhancing the semantic understanding of legal texts (see Fig. 2):

1. **Embeddings Generation:** The embeddings generation process is integral to the RAG architecture. Embeddings are dense vector representations that capture the semantic meaning of text, enabling sophisticated searches based on semantic similarity rather than mere keyword matches [13, 16].

³ <https://developers.google.com/custom-search/v1/overview>.

⁴ <https://huggingface.co/cross-encoder/nli-MiniLM2-L6-H768>.

Table 1. Dataset distribution by source. Authority sources are well-known, credible websites not managed by the DRC government, contributing the largest number of documents. Unofficial sources are independent or privately run websites that also provide legal information. There are no documents from official sources directly managed by the DRC state, highlighting a reliance on non-governmental sources for legal data.

Source	Type	Quantity (documents)
leganet.cd	authority	2690
drointcongolais.info	authority	2678
leganews.pro	authority	2590
legalrdc.com	unofficial	211
village-justice.com	unofficial	161
droint-afrique.com	unofficial	78

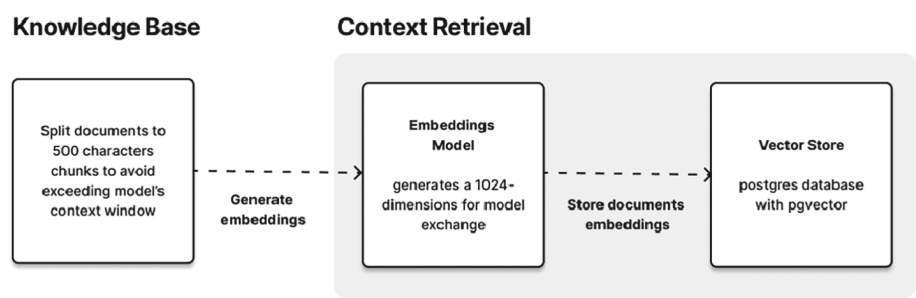


Fig. 2. Context Retrieval Layer

We processed each document by splitting it into chunks of 500 characters to adhere to the context length limitations of various embedding models (see Table 2). This approach ensures that embeddings are generated for manageable text segments, maintaining semantic coherence.

2. **Storage:** The generated embeddings are stored in a PostgreSQL⁵ database with the **pgvector**⁶ extension. This storage solution supports efficient and scalable management of high-dimensional vectors, facilitating quick retrieval during the search process.

We chose an embedding size of 1,024 dimensions to ensure flexibility and compatibility with various models, avoiding overfitting and ensuring the embeddings are versatile across different use cases.

⁵ <https://www.postgresql.org/>.
⁶ <https://github.com/pgvector/pgvector>.

Table 2. Embedding Models Overview—This table lists the embedding models utilized in our study. Models accessed via **API** represent proprietary solutions requiring online access, while **Weights** indicate open-source models that can be used locally through Ollama.

Provider	Model	Access	Context Length
Mistral	mistral-embed	API	512
OpenAI	text-embedding-3-small	API	8191
OpenAI	text-embedding-3-large	API	8191
OpenAI	text-embedding-ada-002	API	8191
Google	text-embedding-004	API	2048
Ollama	mxbai-embed-large	Weights	512
Ollama	nomic-embed-text	Weights	8192

4.3 Response Generation Layer

This layer combines retrieval and generation processes to provide contextually relevant and accurate responses to user queries:

1. **Documents Retrieval:** This process starts with the user’s query, this query is then used to perform a semantic search across the document corpus, leveraging the embeddings stored in the previous layer. The retrieval process involves calculating vector similarity using cosine similarity [4], which measures the closeness of document embeddings to the query. The top **K** documents ($K = 5$ in our setup) are initially retrieved based on their semantic similarity.
2. **Re-Ranking:** The retrieved documents undergo a **Re-Ranking** step, where their relevance is reassessed based on semantic similarity and contextual significance. This step ensures that the returned documents are the most relevant and contextually accurate.
3. **Response Generation:** The final step involves generating a response based on the re-ranked documents. This architecture ensures that the generated response is both semantically accurate and contextually appropriate, providing users with relevant legal information along with proper citations to the original sources.

A key enhancement in our methodology is the integration of a System Prompt, which provides initial instructions guiding the LLM’s behavior throughout interactions. Specifically, we instruct the LLM to communicate in simplified language and an accessible tone tailored to the DRC’s legal context. Additionally, to prevent hallucinations, the LLM is programmed to refrain from generating answers when relevant context is lacking. In such cases, it responds with a message like, *“I’m sorry, I cannot assist due to insufficient information on this subject.”* This ensures that the LLM avoids guessing, restricts its responses to strictly legal topics, and mitigates risks associated with misuse or prompt manipulation.

By integrating these layers our methodology creates a robust system for delivering precise and contextually relevant legal information specific to the DRC’s legal landscape.

5 Experiments

Our experimental evaluation commenced with zero-shot question-answering tasks using existing foundation models. This preliminary phase aimed to accomplish two objectives: selecting the most appropriate base model for Juro and benchmarking state-of-the-art models within the context of the DRC legal system.

5.1 Settings

For model inference, we utilized Ollama⁷, a versatile tool that allows seamless interaction with open-source models. API calls for proprietary models. The LLM settings were carefully calibrated to ensure consistent performance across different models. Specifically, the **temperature** was set to **0.7** for each model to maintain a balance between creativity and coherence in the responses. The **top_p** parameter, which controls the nucleus sampling, was configured to **1** whenever adjustable, ensuring that the models considered the full probability distribution during generation [6]. The model input consisted of an **instruction** followed by a **query**.

5.2 Evaluations

For this evaluation, we employed the 2022 DRC Magistrate Test, which comprises a set of 14 legal questions (see Fig. 3). While this set of questions is limited and does not provide enough data for a statistically robust analysis, it offers preliminary insights into the inherent capabilities of different foundational models in handling legal queries specific to the Congolese legal context.

These questions were chosen due to their relevance and representativeness of the legal challenges within the DRC. Each question was manually evaluated by 18 local legal experts, a process that, while time-consuming, provided valuable qualitative feedback. The evaluations provided by these experts were then averaged to derive a final accuracy score, expressed as a percentage (see Table 3). This score served as a critical metric in determining the most suitable base model for Juro.

6 Results

6.1 Hallucinations

When reviewing the answers, we observed that the foundational language models often produced responses that aligned with Congolese legal principles, due to the

⁷ <https://ollama.com/>.

1	question	category
2	Peut-on constituer un prévenu gardien d'un objet saisi ? Da...	procédure pénale
3	Les expressions, auteur présumé de l'infraction, l'inculpé,...	procédure pénale
4	Quelle est la différence entre l'amnistie et la grâce du po...	procédure pénale
5	Le ministère public peut-il requérir devant le juge le clas...	procédure pénale
6	Le ministère public peut-il aussi introduire la procédure d...	procédure pénale
7	La nationalité congolaise peut-elle être détenue concurremm...	droit civil des personnes
8	La dissolution du mariage par les autorités coutumières ou ...	droit civil des personnes
9	Quelles sont les trois formes de testament consacré par la ...	droit civil des personnes
10	Monsieur FULANI marié à Madame SONGOLO décède en laissant d...	droit civil des personnes
11	Comment qualifie-t-on dans leur ordre progressif de degré d...	droit pénal général
12	Un condamné incarcéré à 12 heures du matin, pour subir un j...	droit pénal général
13	En République Démocratique du Congo, la peine de fouet qui ...	droit pénal général
14	Le calcul de jour de détention d'une personne incarcérée\ n...	droit pénal général
15	En quoi l'amende judiciaire est différente de l'amende tran...	droit pénal général

Fig. 3. A structured evaluation question set categorizing factual legal questions by law field, focusing on questions with answers directly found in legal texts. This primary dataset excludes reasoning-based questions to streamline and facilitate human evaluation.

influence of French, Belgian, and international law on Congolese law. However, while these answers could be correct in the context of French law, they were only partially applicable or incorrect for the Congolese context. This usually resulted in responses that were close to correct but included outdated or imaginary legal references, likely due to the models’ greater familiarity with French legal sources than with specific Congolese legislation (see Fig. 4, 5). For example, in response to the question *“Which ordinance governs digital law in the DRC?”*, both GPT-4-Turbo and Juro provided answers. However, GPT-4-Turbo cited an incorrect legal reference, while Juro correctly identified the applicable legal reference: *“Ordinance-Law No. 23-010 of March 13, 2023.”* This highlights the subtle nature of hallucinations in generated answer and the importance of expert review for accuracy.

6.2 Performance Improvement

Based on the evaluation results, **gpt-4-turbo** demonstrated the highest accuracy, achieving a score of **65.3%**, and was therefore chosen as the foundational model for Juro. To assess Juro’s effectiveness, we conducted a zero-shot question-answering evaluation using the same set of 14 questions from the 2022 DRC Magistrate Test. The generated answers were then reviewed by the same panel of 18 local legal experts, with the accuracy score calculated by averaging their assessments.

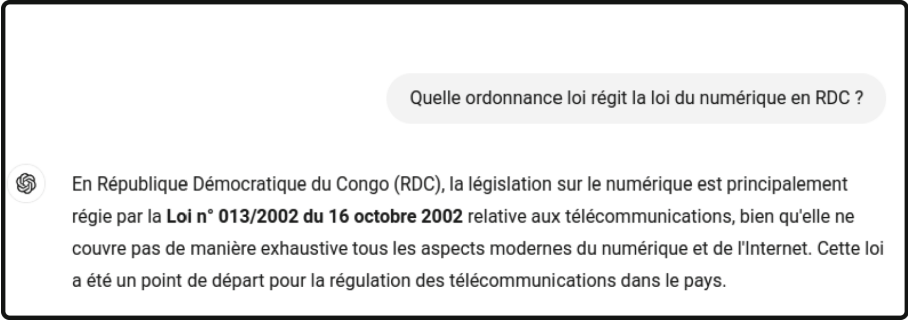


Fig. 4. INCORRECT: In the Democratic Republic of Congo (DRC), the legislation on digital matters is primarily governed by **Law No. 013/2002 of October 16, 2002**, relating to telecommunications, although it does not exhaustively cover all modern aspects of digital technology and the Internet. This law served as a starting point for the regulation of telecommunications in the country.

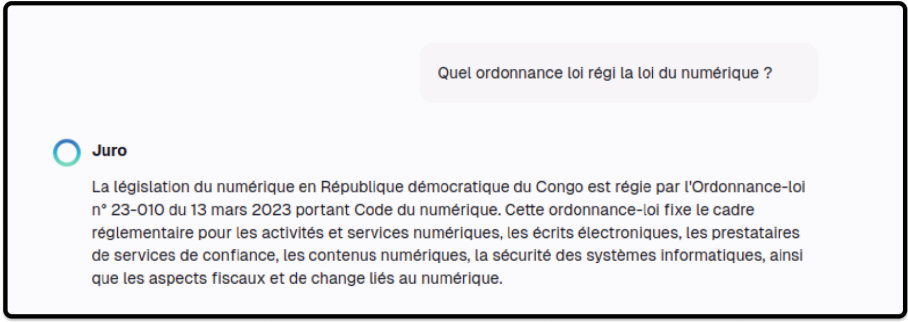


Fig. 5. CORRECT: In the Democratic Republic of Congo, digital legislation is governed by **Ordinance-Law No. 23-010 of March 13, 2023**, establishing the Digital Code. This ordinance-law sets the regulatory framework for digital activities and services, electronic writings, trust service providers, digital content, cybersecurity, and digital-related tax and exchange aspects.

Table 3. Evaluation results of existing models—This table presents the accuracy scores of various LLMs when evaluated using the 2022 DRC Magistrate Test. The models were assessed in a zero-shot context, with no prior fine-tuning.

Model	Access	Parameters	Score
gpt-4-turbo	API	-	65.3%
gpt-3.5-turbo	API	-	65.2%
gemini-pro	API	-	59.6%
mistral-large-latest	API	-	58.4%
llama2	Weights	7b	43.1%
mistral	Weights	7b	38.0%
vicuna	Weights	7b	36.4%

1. **Performance Improvement:** We observed a **20%** increase in Juro’s accuracy score, reaching **85.8%**. This significant improvement highlights the importance of integrating local Congolese context into the model, which enhances both the quality and relevance of its responses.
2. **Knowledge Base Limitations:** Our analysis revealed that Juro’s incorrect answers typically stemmed from a lack of relevant documents in the knowledge base. This underlines the necessity of incorporating a more comprehensive set of Congolese legal texts to improve Juro’s accuracy and reliability. However, integrating the entire body of Congolese law presents challenges due to the resource-constrained setting we are working in. Expanding the knowledge base requires considerable resources for acquiring, processing, and indexing legal documents—resources that are currently limited. Nonetheless, expanding the legal corpus remains a top priority to further improve Juro’s capability to provide precise and contextually accurate responses.

7 Conclusion

This paper introduced **Juro**, a Retrieval-Augmented Generation (RAG) AI chatbot tailored to enhance legal information accessibility in resource-constrained environments such as the Democratic Republic of Congo (DRC). The development of **Juro** addresses critical challenges in accessing legal information in settings where digital infrastructure is scarce, legal systems are complex, and reliable information is often difficult to obtain. By leveraging the power of Large Language Models (LLMs) and a specially curated corpus of over 8,400 legal documents, **Juro** has demonstrated its capacity to simplify legal language and provide accurate legal information while ensuring citation reliability.

The key contributions of this work include the development of the first LLM-based legal chatbot tailored to the Congolese legal context, the construction of a comprehensive legal corpus specific to the DRC, and empirical evidence demonstrating **Juro**’s superior performance in legal question-answering tasks compared to existing models. By integrating context retrieval, **Juro** represents a significant step forward in the deployment of AI-driven solutions for legal information in developing countries, where access to justice is often hindered by a lack of resources and information.

Despite the promising results, there are several avenues for future research. First, further improvements to the underlying LLM models could be achieved through fine-tuning with additional domain-specific data, particularly focusing on local legal nuances and regional linguistic variations. Second, expanding the RAG architecture to include real-time updates of legal documents would ensure that **Juro** remains current with the latest legal changes and rulings. Lastly, the implementation of more robust validation techniques, such as user studies with legal professionals and citizens, would provide deeper insights into **Juro**’s practical application in real-world scenarios.

In conclusion, **Juro** provides a solid foundation for the integration of AI in legal systems within low-resource settings. Its flexibility, scalability, and adaptability across various contexts suggest that it has the potential to be replicated in other developing countries facing similar information access challenges. Future research in this domain should focus on enhancing the legal chatbot's capabilities while ensuring the ethical and fair use of AI in justice systems.

References

1. Chalkidis, I., Androutsopoulos, I., Aletras, N.: Neural legal judgment prediction in English (2019). <https://arxiv.org/abs/1906.02059>
2. Colombo, P., et al.: Saullm-7b: a pioneering large language model for law (2024). <https://arxiv.org/abs/2403.03883>
3. Dai, Y., et al.: Laiw: a Chinese legal large language models benchmark (2024). <https://arxiv.org/abs/2310.05620>
4. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.* 41, 391–407 (1990). <https://api.semanticscholar.org/CorpusID:3252915>
5. Touvron, H., et al.: Llama 2: open foundation and fine-tuned chat models (2023). <https://arxiv.org/abs/2307.09288>
6. Fei, Z., et al.: Lawbench: benchmarking legal knowledge of large language models (2023). <https://arxiv.org/abs/2309.16289>
7. He, Z., et al.: Simucourt: building judicial decision-making agents with real-world judgement documents (2024). <https://arxiv.org/abs/2403.02959>
8. Huang, Q., et al.: Lawyer llama technical report (2023). <https://arxiv.org/abs/2305.15062>
9. Kien, P.M., Nguyen, H.-T., Bach, N.X., Tran, V., Nguyen, M.L., Phuong, T.M.: Answering legal questions by learning neural attentive text representation. In: Scott, D., Bel, N., Zong, C. (eds.) *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 988–998, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. <https://doi.org/10.18653/v1/2020.coling-main.86>. <https://aclanthology.org/2020.coling-main.86>
10. Lai, J., Gan, W., Wu, J., Qi, Z., Yu, P.S.: Large language models in law: a survey (2023). <https://arxiv.org/abs/2312.03718>
11. Lewis, P.: Retrieval-augmented generation for knowledge-intensive nlp tasks (2021). <https://arxiv.org/abs/2005.11401>
12. Luo, B., Feng, Y., Xu, J., Zhang, X., Zhao, D.: Learning to predict charges for criminal cases with legal basis. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics (2017). <https://doi.org/10.18653/v1/d17-1289>
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013). <https://arxiv.org/abs/1301.3781>
14. Nguyen, H.-T.: A brief report on lawgpt 1.0: A virtual legal assistant based on gpt-3 (2023). <https://arxiv.org/abs/2302.05729>
15. OpenAI. Gpt-4 technical report (2024). <https://arxiv.org/abs/2303.08774>
16. Global vectors for word representation: Pennington, J., Socher, R., Manning, C.D. Glove. In: *EMNLP* 14, 1532–1543 (2014)
17. Pipitone, N., Alami, G.H.: Legalbench-rag: a benchmark for retrieval-augmented generation in the legal domain (2024). <https://arxiv.org/abs/2408.10343>

18. Smith, R.W.: An overview of the tesseract ocr engine. In: Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 2, pp. 629–633 (2007). <https://api.semanticscholar.org/CorpusID:7038773>
19. Yue, S., et al.: Disc-lawllm: fine-tuning large language models for intelligent legal services (2023). <https://arxiv.org/abs/2309.11325>
20. Zhou, Z., et al.: A chinese legal knowledge-enhanced large language model (2024). <https://arxiv.org/abs/2406.04614>



Campus WiFi Demand Prediction: A Case Study on KNUST Campus

Justice Owusu Agyemang^(✉), Manhyia Gideon Aboagye,
Awumee Gabriel Selorm, Somuah Obed, Addai Clinton,
and Henyo Chester Elikplim

Faculty of Electrical/Computer Engineering, Kwame Nkrumah University of Science
and Technology, Kumasi, Ghana

`jay@knust.edu.gh,`

`{agmanhyia1,sgawumee,oksomuah,caddai7,cehenyo}@st.knust.edu.gh`

Abstract. The increased student population at Kwame Nkrumah University of Science and Technology (KNUST) has resulted in significant congestion and slow internet connectivity on campus. We propose a robust model for predicting Wi-Fi demand on the KNUST campus to address this issue. Using time series data collected from the University Information Technology Services (UITs), we evaluate three forecasting models—Auto-Regressive Integrated Moving Average (ARIMA), Gradient Boosting Regressor (GBR), and Extreme Gradient Boosting (XGBoost)—to determine the optimal approach for predicting Wi-Fi traffic demand in both faculty and hall areas. The results indicate that the GBR model performed best in the faculty area, achieving a Mean Absolute Percentage Error (MAPE) of 18.73%, while the XGBoost model excelled in the hall and commercial areas with a MAPE of 17.95%.

Keywords: Wi-Fi Demand Prediction · Neural Network (NN) · ARIMA · GBR · XGBOOST · Time Series Forecasting

1 Introduction

The internet has undergone significant development in recent years due to the growth of the global population, the proliferation of smart devices, and the emergence of new technologies. Businesses, banking, and e-commerce sectors demand highly secure and seamless network platforms to remain competitive in the job market.

Internet traffic refers to the data, bits, and packets transmitted across devices on a computer network at any given time. Due to its temporal nature, data collected at uniform intervals, referred to as time series data, is analyzed using mathematical models to predict future values based on past observations.

Recurrent Neural Networks (RNN) and Auto-Regressive Integrated Moving Average (ARIMA) are widely used mathematical models for time series forecasting. ARIMA is particularly effective for certain types of time series. However,

one limitation of ARIMA is its reliance on stationarity, making it less suitable for modeling nonlinear time series. In contrast, RNN and XGBoost have been shown to outperform ARIMA in predicting nonlinear time series. The advent of smart homes, cities, and buildings has further increased the demand for internet resources. For efficient management and resource allocation, Internet Service Providers (ISPs) must understand Internet demand trends at specific times.

At Kwame Nkrumah University of Science and Technology (KNUST), two ISPs—MTN and Vodafone—provide internet services. MTN manages connectivity in the halls and commercial areas, while Vodafone handles faculty areas. The current Wi-Fi infrastructure, implemented in 2015, was designed for a smaller student population. Over the years, the student population has grown from 23,591 in 2011 to over 85,000 in 2021, leading to increased congestion and slow internet speeds on campus. For ISPs to effectively manage network resources, they require accurate demand predictions.

This study seeks to model Wi-Fi demand patterns on the KNUST campus by collecting and analyzing traffic usage data. Machine learning and deep learning algorithms, such as ARIMA, RNN, and XGBoost, are applied to identify patterns and forecast future network traffic. Key performance indicators (KPIs) are used to evaluate the effectiveness of the models.

The remainder of this paper is organized as follows: Sect. 2 reviews related work and sets the study objectives. Section 3 outlines the methodology, including data collection and model training. Section 4 presents the results and discusses the findings. Finally, Sect. 5 concludes the paper and provides recommendations for future work.

2 Related Works

Internet traffic is in high demand due to emerging technologies and an increase in the number of users. This has made demand prediction vital in both business and daily life. This section discusses previous research related to internet traffic prediction.

Recent advancements in network traffic prediction have relied on the analysis of long-term datasets, with different time spans playing a crucial role in model training and evaluation. Jiang et al. [1] used a public dataset collected over six months containing internet traffic volume from the backbone network. The Deep Neural Network (DNN) and several baseline models were considered during the experiment, where the DNN demonstrated lower Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), making it the preferred model. Aung et al. [2] used a public dataset containing the source and destination IP addresses, ports, inter-arrival times, and layer-7 protocols (Applications) for each flow in their experiment. A machine learning algorithm supporting dynamic QoS policies was applied to predict network traffic demand, achieving an accuracy of 98.97%.

Dalgkitsis et al. [3] utilized four months of subscriber traffic data provided by Vodafone, containing throughput values from random base stations. The Auto-Regressive Integrated Moving Average (ARIMA) and Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) models were used to train the

dataset. The LSTM and RNN models were preferred over ARIMA due to superior performance during evaluation. A one-year dataset from an ISP, containing 1657 samples from the Time Series Data Library, was also studied. The dataset included attributes such as traffic load and inter-arrival time values. Three different Deep Neural Network architectures achieved an RMSE of 0.028 [4]. Rutka et al. [5] used a one-year dataset from a public site. The study emphasized self-similar traffic prediction using neural networks, specifically the Multi-Layer Perceptron Network (MLP) and Radial Basis Function Network (RBFN). The MLP was chosen as the best model for traffic prediction due to its ability to handle self-similarity effectively.

Chen et al. [6] conducted a survey that classified machine learning applications into short- and long-term predictions. The exploration included techniques such as neural networks, support vector machines, linear regression, and principal component analysis. Findings suggested that neural networks were most frequently used in network traffic prediction. Maheshwari et al. [7] also conducted a survey to measure traffic needs and designed a Hidden Markov Model-based forecasting algorithm, which was validated for different packet sizes.

Yang et al. [8] proposed a network traffic identification approach that integrates machine learning with Deep Packet Inspection (DPI). DPI was used to identify known and specific application traffic, while machine learning handled encrypted and unknown traffic that DPI alone could not classify. However, due to DPI analyzing the content of every packet in real-time, this approach can be computationally intensive in high-speed networks with significant traffic volumes. Kuznietsova et al. [9] addressed data mining methods in two ways: predicting internet traffic volume and classifying clients based on their roaming behavior. Neural networks, gradient boosting, random forest, and support vector machines were explored. Gradient boosting was selected for its superior accuracy and completeness.

Rishabh et al. [10] employed Discrete Wavelet Transform (DWT), ARIMA, and RNN-based techniques to model time series data recorded per hour, day, and minute, created by R.J. Hyndman [11]. Ultimately, the DWT model was preferred due to its ability to capture both linear and non-linear aspects of the time series using high- and low-pass filters. For short-range network traffic prediction, I. Kļevecka [12] used a time series model combining neural networks and ARIMA methods. Data was collected from telephone and IP packet-switching networks. Metrics such as RMSE, mean absolute error (MSE), and mean absolute percentage error (MAPE) were employed to evaluate ARIMA's effectiveness. Neural networks outperformed prior methodologies, yielding superior forecasts for short-range networks.

A hybrid model combining linear time series ARIMA with non-linear Multilayer Artificial Neural Networks was proposed to predict immediate network traffic. The study used data obtained every eight minutes from the Guangzhou network center's web server throughout the day [13, 14].

3 Methodology

This section outlines how we conducted our study. It explains the steps we took and the methods we used to answer the research questions stated in Sect. 2.

3.1 Data Collection and Preprocessing

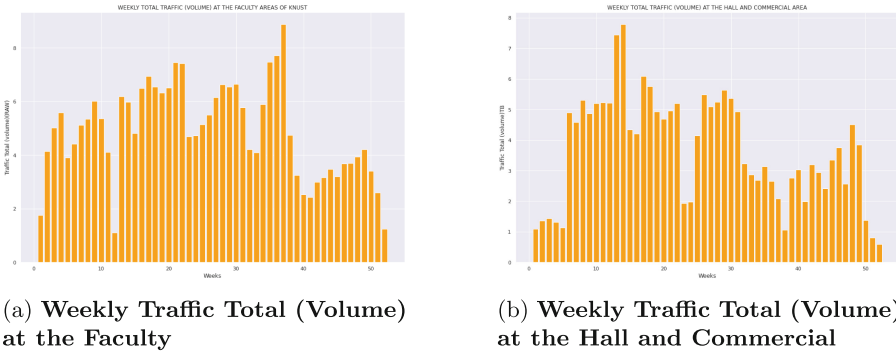


Fig. 1. Comparison of Weekly Traffic Totals at Different Locations

The internet traffic dataset was taken from the University Information Technology Services (UITS). The time series dataset consists of a total of 362 time series data samples that were recorded daily from two ISPs (MTN) and Vodafone). The MTN ISP as stated takes care of services in the halls and commercial area whilst Vodafone takes care of services at the various faculties. Figure 3 and 4 show the total traffic in the university’s backbone network from August 2021 to August 2022. Figure 3 records Wednesdays, Thursdays, and Fridays as the high-traffic days with 16% for faculty areas. Figure 3 records Wednesdays and Thursdays with 16% for the Halls and Commercial Areas. Both Fig. 3 and 4 record weekends as the least traffic days.

3.2 Exploratory Analyses on Data

The dataset contains a total of 13 attributes. These include; Date Time, Date Time (RAW), Traffic Total (volume)TB, Traffic Total (volume)(RAW), Traffic Total (speed)Mbit/s, Traffic Total (speed)(RAW), Traffic In (volume)TB, Traffic In (volume)(RAW), Traffic In (speed)Mbit/s, Traffic In (speed)(RAW), Traffic Out (volume)TB, Traffic Out (volume)(RAW), Traffic Out (speed)Mbit/s, Traffic Out (speed)(RAW) values recorded each day. The dataset was divided into 2 sets of training and testing with different splitting ratios for different Machine learning models. The dataset was cleaned to ensure that there were no missing values. It was then divided into two sets, a training set and a testing set. The

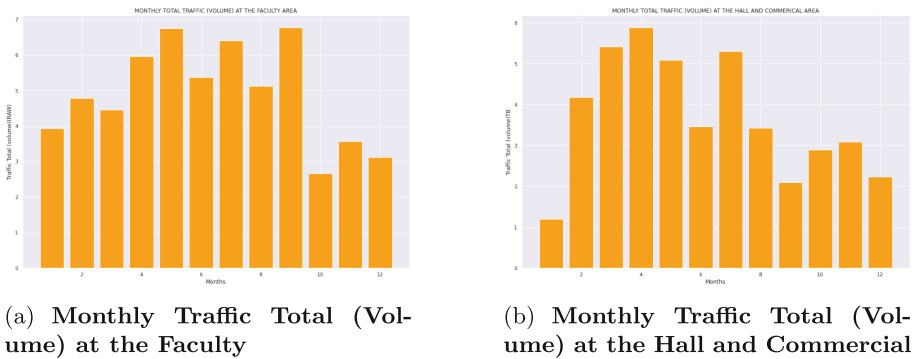


Fig. 2. Comparison of Monthly Traffic Totals at Different Locations

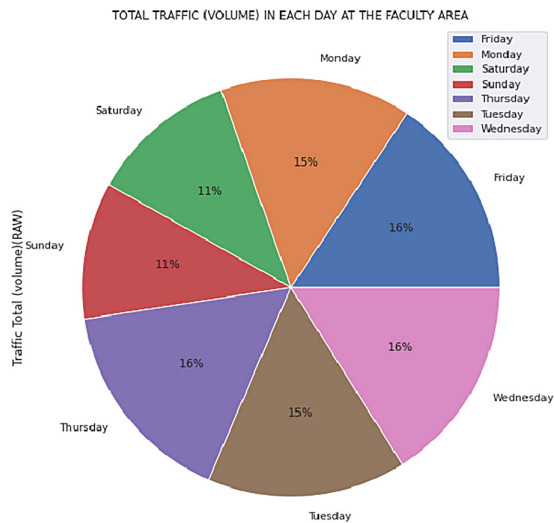


Fig. 3. Daily Traffic Total (Volume) at the Faculty

training set was 80% of the total dataset, and the testing set was 20% of the total dataset to train all the different models.

Certain exploratory analyses were done on the data to check the total traffic in days, weeks, and months in both the halls and commercial area, and faculty areas. From Figs. 1a and 1b, the 37th week happens to have the highest recorded total traffic in the faculty area and that of the hall and commercial area in the 14th week. Figures 2a and 2b also show that the 5th and 9th months that is May and September happened to have the highest recorded total traffic in the faculty areas as expected. For the hall and commercial area, the 4th month which is April happened to be the highest.

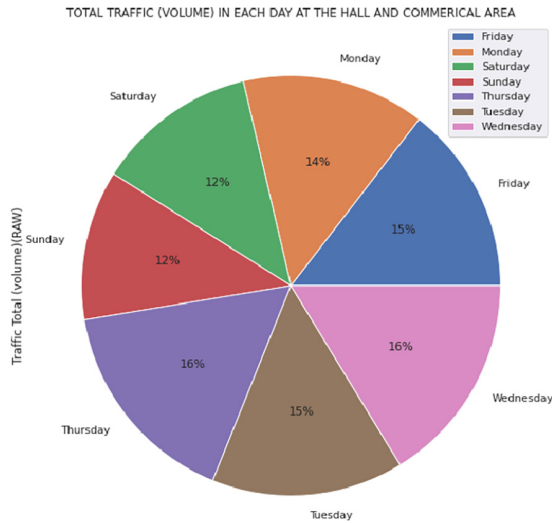


Fig. 4. Daily Traffic Total (Volume) at the Hall and Commercial

3.3 Model Training

Autoregressive Integrated Moving Average (ARIMA) is mostly used in time series analysis. It analyzes the provided data to find the pattern or drift in the data and projects roughly the next values to support better decision-making in the future. Based on its prior values, the ARIMA model makes predictions for a given time series. ARIMA is a model in which a variable changes and regresses on its own lagged or past value and looks for differences between current values and historical values in static data. Typically, an ARIMA model is commonly built using the notation ARIMA (p, d, q). p is the number of lag observations required to train the model (i.e. lag order). d is the number of times differencing is applied (i.e. degree of differencing), and q is known as the size of the moving average window (i.e. order of moving average) [15]. The objective is to obtain stationary data that is not seasonal. This indicates that the data series' statistical characteristics, such as mean, variance, and autocorrelation, remain constant across time (Fig. 5).

The stationarity of the data was first checked by the Augmented Dickey-Fuller test (ADF), an essential step when analyzing time series. The ARIMA models presume the series is stationary (i.e., its statistical properties like mean and variance do not change over time); hence, the series was checked by the ADF test to see if it meets this prerequisite. It gave a very high p-value, hence proving that the dataset was not stationary. Because of this, differencing had to be performed on the data by subtracting the current value from the previous one. This managed to make the series stationary, verified by running an ADF test a second time. With stationarity proven, Autocorrelation (ACF) and partial autocorrelation (PACF) plots were obtained for the appropriate values of lagging

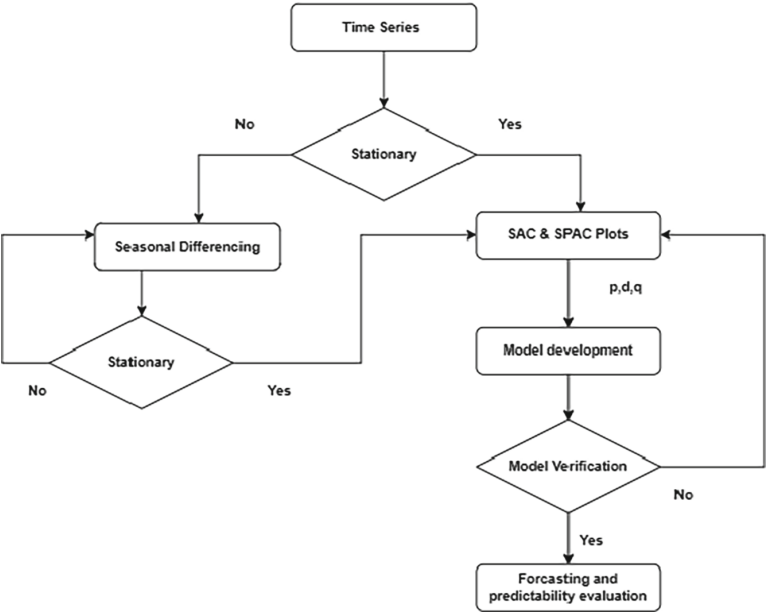


Fig. 5. ARIMA Architecture

in determining ARIMA model parameters. These plots are used to determine the number of lag observations for both the autoregressive and the moving average parts, which, if used in the model, will give an accurate forecast.

The ARIMA model was then developed based on the ACF and PACF plots, the model for the faculty area was configured with an order of (1, 1, 4) where as that of the hall and commercial area was (5, 1, 4) where 1 or 5 represents the number of AR terms, 1 the level of differencing, and 4 the number of MA terms. The model was trained on the data and fitted using historical traffic volume. The model’s performance was evaluated using Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).

Gradient boosting Regressor (GBR) is a type of ensemble supervised machine learning algorithm that leverages the strength of several weak predictive models to generate a powerful predictive model [16]. In this study, we employed a GBR model to predict the total traffic volume in the Faculty area.

The dataset includes relevant features such as traffic volume and engineered lag features, which provide a comprehensive view of the temporal dynamics in the traffic data. Missing values were handled using forward filling, a technique that replaces missing entries with the preceding value to ensure continuity in the time series data. To maintain the temporal sequence, the data was sorted by the ‘Date Time’ column, ensuring that the predictions were consistent with the order of events.

Feature engineering was used to capture trends and seasonality in the traffic volume. Lag features were created by shifting the total traffic volume by one, two, and three time steps (denoted as **Traffic_lag1**, **Traffic_lag2**, and **Traffic_lag3**), allowing the model to consider the immediate past traffic volumes. A rolling mean feature was computed over a window of three-time steps to smooth out short-term fluctuations and highlight longer-term trends.

After creating the necessary features, rows containing NaN values introduced by the lagging process were dropped to ensure clean input data for the model. The data was then split into training and test sets, with 80% of the data used for training and 20% for testing.

The model was initiated with 1000 estimators, a learning rate of 0.1, and a maximum tree depth of 3 to avoid possible under or overfitting. The model was trained on the training set, using the past traffic data as input to predict the future traffic volume. During the training phase, the GBR algorithm iteratively improved its predictions by focusing on reducing the error from previous iterations. This entire process made it possible for the model to progressively refine its understanding of the underlying patterns in the traffic data. The model's performance was evaluated using RMSE and MAPE.

Finally, Extreme gradient boosting, or XGBoost is a gradient-boosting implementation. It is swift and performs well. A scalable solution for learning the Tree ensemble approach is XGBoost. It is employed for a variety of purposes and also supports outdoor memory. Its parallel calculation method makes it faster than other boosting algorithms. The fact that XGBoost is scalable in nature accounts for its superior performance. It performs better than gradient boosting because of parallel processing. It manages the issue of overfitting. On many datasets, it produces higher performance results. It is essentially a tree-building algorithm. It is utilized for ranking, regression, and classification with unique loss functions (Fig. 6).

The dataset for the model training was first preprocessed to put it in a form that can be used. The **Date Time** column was converted to a datetime format, and missing values were handled through forward-filling. The **Traffic Total (volume)** column was also cleaned by removing non-numeric characters and then converted to a numeric format for modeling.

Feature engineering techniques were employed to model the data. The techniques involved creating lag features (**Traffic_lag1**, **Traffic_lag2**, **Traffic_lag3**) to capture past traffic volumes, and a rolling mean feature (**Traffic_roll_mean**) to represent recent traffic trends.

The dataset was split into an 80–20 training and test ratio and the XGBoost Regressor was selected as the modeling algorithm. This was initialized with 1000 estimators, a learning rate of 0.1, and a maximum tree depth of 1. Additionally, Early stopping was implemented to prevent overfitting. This is done by stopping training if the test performance fails to improve for 10 consecutive iterations.

The model's performance was evaluated using RMSE and MAPE.

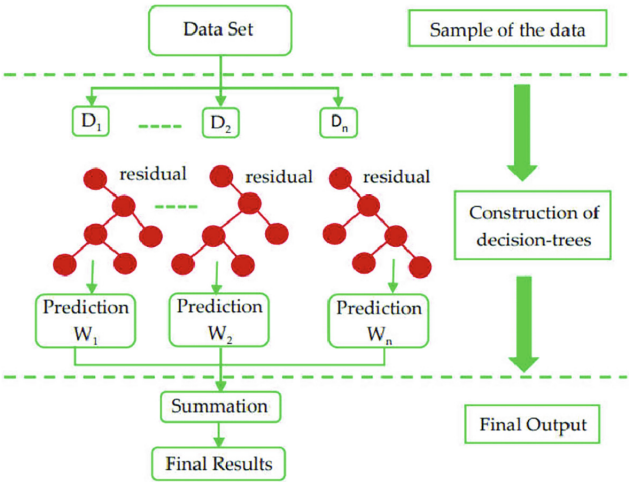


Fig. 6. XGBoost Architecture [17]

4 Results and Discussion

The experiments were conducted using two different datasets from the University’s Information Technology Service (UITS). These datasets record the amount of total traffic received by the two ISPs: MTN and Vodafone. In this section, we present and demonstrate the results of fitting an available dataset using three techniques: ARIMA, Gradient Boosting Regression (GBR), and Extreme Gradient Boosting (XGBoost).

4.1 Auto-Regressive Integrated Moving Average (ARIMA)

Figures 7 and 8 show the one-year forecast of total traffic volume in the Faculty areas and the Hall and commercial area on KNUST Campus by fitting the ARIMA(2,1,4) model to the time series data. The Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) for the Faculty and Hall areas were 1.94 and 25.44%, and 2.29 and 30.82%, respectively.

4.2 Gradient Boosting Regression (GBR)

Figs. 9 and 10 show the traffic volume forecast for a given period at the Faculty areas and the Hall and commercial area on KNUST Campus using the GBR model. For the Hall and commercial area dataset, an 80%–20% train-test split was employed. The RMSE and MAPE for the Faculty and Hall areas were 1.22 and 18.73%, and 0.85 and 21.63%, respectively.

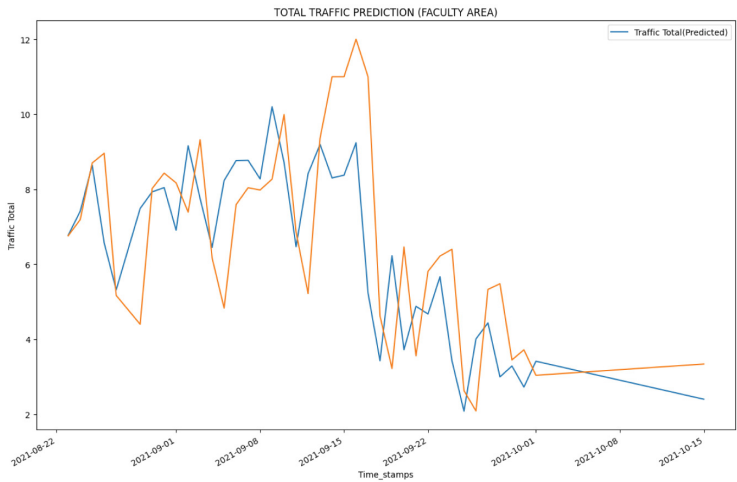


Fig. 7. ARIMA Prediction Results at the Faculty Area

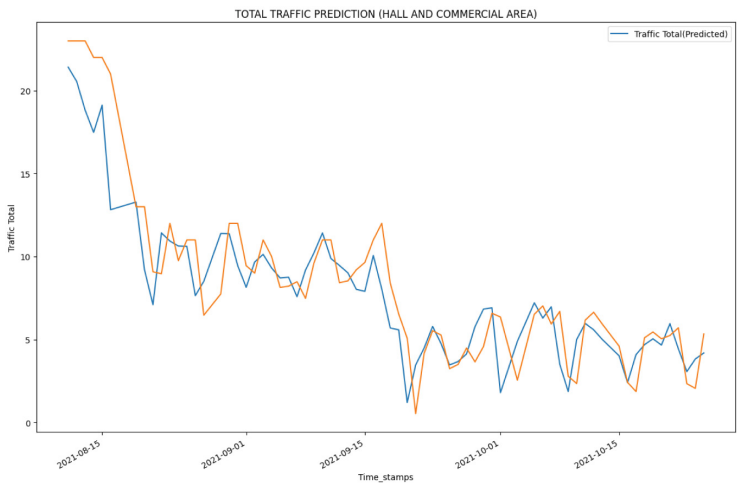


Fig. 8. ARIMA Prediction Results at the Hall and Commercial Area

4.3 Extreme Gradient Boosting (XGBoost)

Figures 11 and 12 show the two-week forecast of total traffic volume in the Faculty areas and the Hall and commercial area on KNUST Campus using the XGBoost model. The RMSE and MAPE for the Faculty and Hall areas were 1.42 and 24.72%, and 2.91 and 17.95%, respectively.

The performance of the various models is summarized in Table 1. The GBR model demonstrated the best overall performance, particularly in the Faculty area, where it recorded the lowest MAPE of 18.73% and RMSE of 1.22. The

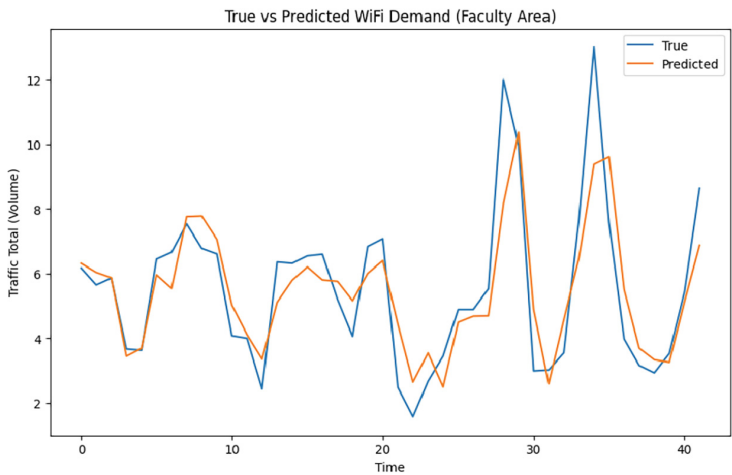


Fig. 9. GBR Prediction Results at the Faculty Area

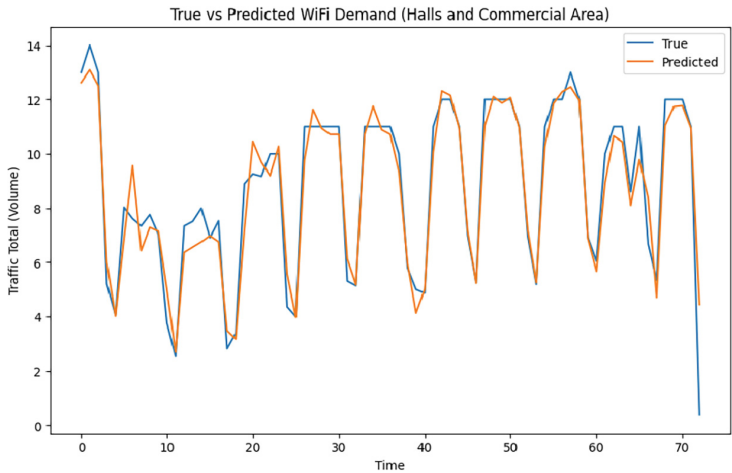


Fig. 10. GBR Prediction Results at the Hall and Commercial Area

model’s ability to capture complex patterns and avoid overfitting, coupled with its sensitivity to outliers, made it more suitable for the Faculty area’s traffic data.

However, for the Hall and commercial area, the XGBoost model showed superior performance with the lowest MAPE of 17.95%. This suggests that XGBoost was able to handle the Hall area’s larger variations in traffic more effectively than the other models. XGBoost’s capability to manage a large number of features, its regularization techniques, and its ability to prevent overfitting allowed it to outperform both ARIMA and GBR models in this context.

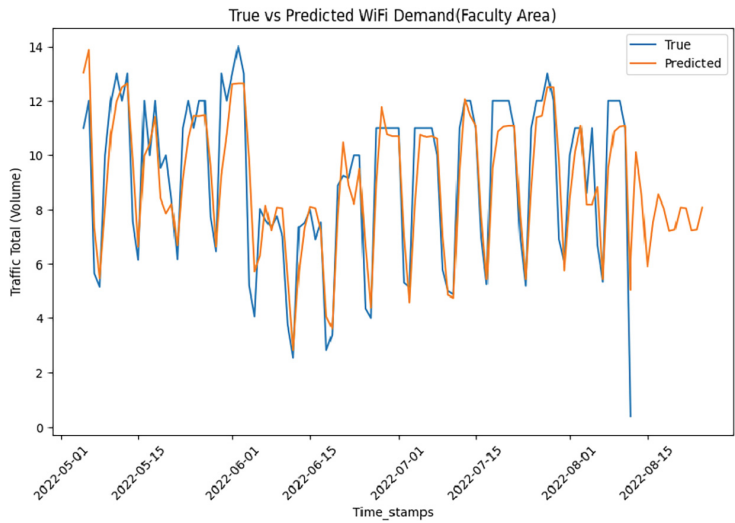


Fig. 11. XGBOOST Prediction Results at the Faculty Area

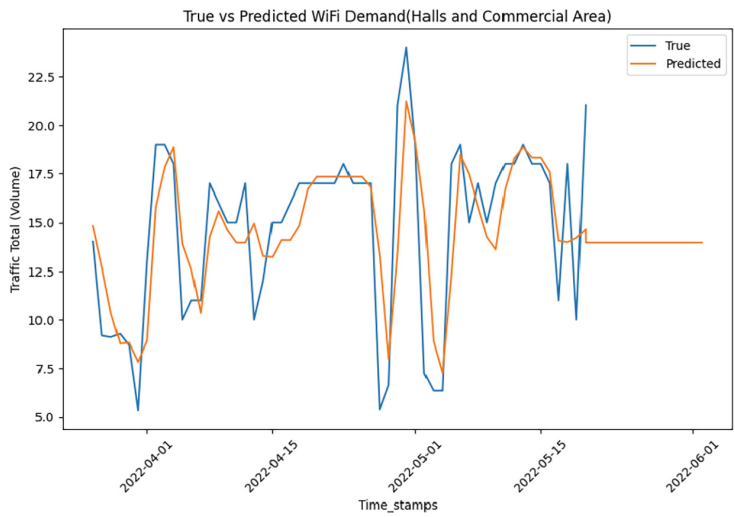


Fig. 12. XGBOOST Prediction Results at the Hall and Commercial Area

Thus, while GBR performed best in the Faculty area, XGBoost excelled in handling the more complex data patterns found in the Hall and commercial area, making it the most adaptable model across both datasets.

Table 1. Prediction Results for Different Models at Various Locations

Model	Location	RMSE	MAPE (%)
ARIMA (1, 1, 4)	Faculty Area	1.94	25.44
ARIMA (5, 1, 4)	Hall and Commercial Area	2.29	30.82
GBR	Faculty Area	1.22	18.73
GBR	Hall and Commercial Area	0.85	21.63
XGBoost	Faculty Area	1.42	24.72
XGBoost	Hall and Commercial Area	2.91	17.95

5 Conclusion

In this paper, a robust model for predicting Wi-Fi demand at KNUST was developed to help UITs allocate resources effectively, addressing the issues of slow connectivity and congestion due to the increasing student population. Three forecasting models—ARIMA, GBR, and XGBoost—were applied to analyze time series data from the university’s two main ISPs, MTN and Vodafone. The three models were trained using the dataset and produced different performances in predicting traffic at the Faculty and Hall areas.

Among the models, the XGBoost model demonstrated the best predictive performance in the Hall and commercial area, with a MAPE of 17.95%, while the GBR model showed the best performance in the Faculty area with a MAPE of 18.73%. These results suggest that GBR is effective for smaller, less variable datasets like the Faculty area, while XGBoost is more adaptable for larger, more complex environments like the Hall and commercial area.

References

1. Jiang, W.: Internet traffic prediction with deep neural networks. *Internet Technology Letters* **5**(2), e314 (2022)
2. Aung, S.T., Thein, T.: Internet traffic categories demand prediction to support dynamic qos. In: 2020 5th International Conference on Computer and Communication Systems (ICCCS), pp. 650–654. IEEE (2020)
3. Dalgkisis, A., Louta, M., Karetsos, G.T.: Traffic forecasting in cellular networks using the lstm rnn. In: Proceedings of the 22nd Pan-Hellenic Conference on Informatics, pp. 28–33 (2018)
4. Narejo, S., Pasero, E.: An application of internet traffic prediction with deep neural network. *Multidisciplinary Approaches to Neural Computing*, pp. 139–149 (2018)
5. Rutka, G.: Neural network models for internet traffic prediction. *Elektronika ir Elektrotechnika* **68**(4), 55–58 (2006)
6. Chen, A., Law, J., Aibin, M.: A survey on traffic prediction techniques using artificial intelligence for communication networks. In: *Telecom*, vol. 2, pp. 518–535. MDPI (2021)
7. Maheshwari, S., Mahapatra, S., Cheruvu, K.: Measurement and forecasting of next generation wireless internet traffic, September 2018

8. Yang, B., Liu, D.: Research on network traffic identification based on machine learning and deep packet inspection. In: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), pp. 1887–1891. IEEE (2019)
9. Kuznietsova, N., Bidyuk, P., Kulinich, A.: Data mining methods for evaluation and forecasting the mobile internet traffic in roaming. In: CMIS, pp. 73–86 (2021)
10. Madan, R., Mangipudi, P.: Predicting computer network traffic: A time series forecasting approach using dwf, arima and rnn. In: 2018 Eleventh International Conference on Contemporary Computing (IC3), pp. 1–5 (2018)
11. Hyndman, R.J.: Time series data library (2024). <https://robjhyndman.com/tsdl/>. Accessed 05 Sept 2024
12. Kłevecka, I.: Forecasting traffic loads: Neural networks vs. linear models. *Comput. Modelling New Technol.* **14**(2), 20–28 (2010)
13. DataCenters.com. China telecom guangzhou huadiwan idc (2024). <https://www.datacenters.com/china-telecom-guangzhou-huadiwan-idc>. Accessed 05 Sept 2024
14. Zeng, D., Xu, J., Gu, J., Liu, L., Xu, G.: Short term traffic flow prediction using hybrid arima and ann models. In: 2008 Workshop on Power Electronics and Intelligent Transportation System, pp. 621–625 (2008)
15. Robert Nau. Arima models for time series forecasting (2024). <https://people.duke.edu/~rnau/411arim.htm>. Accessed 05 Sept 2024
16. Wang, F.-K., Mamo, T.: Gradient boosted regression model for the degradation analysis of prismatic cells. *Comput. Ind. Eng.* **144**, 106494 (2020)
17. Jan, M., et al.: Appraisal of different artificial intelligence techniques for the prediction of marble strength. *Sustainability* **15**, 8835 (2023)

Remote Sensing and Artificial Intelligence



A Particle Swarm Optimization-Long-Short Term Memory (PSO-LSTM) Hybrid Model for Forecasting Global Horizontal Solar Radiation

Blessing Olatunde Abisoye¹ , Yanxia Sun¹ , and Wang Zenghui^{2,3} 

¹ Department of Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg, South Africa

222244434@student.uj.ac.za, ysun@uj.ac.za

² Department of Electrical Engineering, University of South Africa, Johannesburg, South Africa
wangz@unisa.ac.za

³ Center of Artificial Intelligence and Data Science, University of South Africa, Johannesburg, South Africa

Abstract. Electrical energy demands require clean and sustainable energy production. Solar power output as a clean energy depends on solar radiation and other meteorological features. The variability of solar radiation and other meteorological features impaired solar energy output, significantly affecting the economics of intelligent grids and microgrid operation reliability. This study proposed a hybrid system that comprises a particle swarm optimization (PSO) algorithm and Long-Short Term Memory (LSTM). The PSO algorithm was employed to conduct the adaptive parameter adjustment of the LSTM models. The study benchmarked the proposed hybrid PSO-LSTM model with LSTM-based Genetic Algorithm (GA), Grey Wolf Optimization (GWO), and Ant Colony Optimization (ACO) models. Four performance evaluation metrics were utilized to compare forecasting accuracy models: R-Square, RMSE, MAE, and MAPE. Comparing the developed model, PSO-LSTM, R^2 (0.950643), RMSE (60.304873), and MAE (25.442901) with the benchmarked model, it was discovered that the PSO-LSTM is superior. Its full implementation will assist in integrating solar systems into the national grids and enable the stakeholders and policymakers to make effective decisions.

Keywords: Global Horizontal Radiation · Metaheuristic Algorithms · LSTM

1 Introduction

Studies and phenomena have proved that fossil fuel methods of generating electricity have significant adverse environmental impacts. Gaseous emissions, environmental degradation, and global warming are phenomena that increase geometrically. These adverse events, in turn, decrease life expectancy [1]. Other problematic situations associated with fossil fuel electric power plants are the cost of setting up the plant, running

and maintenance costs, and the prediction of the exhaustion of fossil fuel globally [2]. The quest for sustainable electric power generation and supply has resulted in exploring renewable energy sources (RES). This implies that the revolution of electric power generation is taking place in the 21st century.

The adoption rate of solar and wind RES is growing faster because of their benefits. Some benefits are decentralized energy production, low operation cost, energy security and independence, decarbonization, low cost of generation and maintenance, and so on [3]. With these benefits, the RES still exhibit seasonal variation in nature, which impaired their power out. The seasonal variation resulted from their dependability on weather features. Solar energy power output depends on solar radiation and other meteorological features such as temperature, wind speed, and so on [4]. Several studies have been conducted to forecast the seasonal variation of solar RES and to plan, estimate, integrate, and make informed decisions. The existing approaches that have been utilized to forecast the RES are Numerical Weather Prediction (NWP) and statistical and artificial intelligence (AI) methods [5, 6]. The NWP does not require training in historical meteorological data. The major challenge of statistical methods is the inability to handle complex nonlinear problems and large datasets. The AI methods can handle the prediction of complex nonlinear problems and large datasets. However, there are still some grey techniques that will better forecast the variability of solar RES, leading to an increase in accuracy and a decrease in computational cost.

Several studies have adopted AI-based strategies: machine learning (ML), deep learning (DL), and ensemble learning [7]. Some of the applied ML methods are artificial neural networks (ANN) [8], support vector machine (SVM) [9], a fuzzy rule-based method [10], and so on. The DL methods implemented are convolutional neural networks (CNN), gated recurrent unit (GRU), long short-term memory (LSTM), and their variances [6, 11]. Also, the variants of ensemble learning, such as decision tree (DT), random forest (RF), gradient boosting (GB), and so on, have been utilized to forecast RES [12, 13]. All these traditional AI methods achieved some level of accuracy. However, traditional AI methods have some drawbacks, viz, overfitting, lack of interpretability, computational complexity, and sensitivity to hyperparameter tuning. Due to these drawbacks, metaheuristic algorithms were applied in some studies to address some of these drawbacks and further increase the forecasting models' accuracy. The metaheuristic algorithms that have been implemented to forecast solar radiation are genetic algorithm (GA) [14], particles swarm optimization (PSO) [15], grey wolf optimization (GWO), cuckoo search algorithm (CS) [16], salp swarm algorithm (SSA) [17] and so on. This study proposed a hybrid AI model that can forecast Global Horizon Radiation. The model will promote the sustainability of electric energy production globally.

1.1 Related Study

Predictions of weather-related systems, such as solar radiation and wind speed, are stochastic, nonlinear, and complex problems [18]. The advent of intelligent computational algorithms marked the beginning of more accurate solutions with less computational cost. AI methods have been implemented to solve spatiotemporal problems in solar radiation forecasting, such as predictions of global horizontal, diffused, and tilted

radiation horizons and regional radiation. This section presents a review of the existing studies on solar radiation forecasting.

In Ref. [19], Ghimire et al. proposed deep residual networks with a BiLSTM model to forecast solar radiation for 5 min, 15 min, 30 min, and 1 h horizons in Nepalgunj of Nepal. An improved complete ensemble empirical mode decomposition with adaptive noise was introduced as a data decomposition method. The model outperformed the benchmarked models with MAE (9.769 W/m^2) and MAPE (5.657%) for a 5-min horizon. However, the study revealed that as the forecasting horizon increases, the model accuracy decreases. In another study, a hybrid model of CNN and SVR was developed for daily ahead solar radiation forecasting of six solar farms in Queensland, Australia [9]. CNN was employed for feature selection, and the CNN and SVR hyperparameters were tuned with the atom search optimization (ASO) algorithm. The model achieved less error scores of RMSE ($2.172 - 3.305 \text{ MJm}^2$) and MAE ($1.624 - 2.370 \text{ MJm}^2$) compared with the eight benchmarked models. Nonetheless, the real-time implementation of the developed model might cause high computational costs. A hybrid model of the DL algorithm comprised of GRU, LSTM, BiLSTM, and stacked LSTM was developed to predict solar radiation one hour ahead of time in Lake Michigan, Chicago, USA. The developed model was optimized with an improved adaptive differential covariance matrix evolution algorithm (ADCMA) [20]. Autoencoder detected outliers in the time series data, and adaptive multivariate empirical mode decomposition (AMEMD) with a simplex direct search algorithm was used to decompose the data. The model was superior compared with the twenty-six benchmarked algorithms. Conversely, like the model in the Ref. [9], the developed algorithm might exhibit high complexity and computational costs.

A better artificial ecosystem-based algorithm with logistic chaotic mapping consumption task and local search was employed to tune the hyperparameters of temporal convolutional networks (TCN) [21]. The modelled algorithm predicted four months of solar radiation at the University of Minnesota. The dataset was decomposed with optimal VMD. Monthly model errors show that the developed model is superior to the compared models. Nonetheless, the study needs to consider the meteorological features. An RF-PSO model was proposed for a daily ahead forecast of solar radiation in the Big Island of Hawaii; the study compared with three traditional models, viz, RF, DT, and MLP. The performance metrics revealed RMSE (27.74) and MAE (24.45), and the proposed model proved its superiority over the compared model [22]. Moreover, a PSO-SVR algorithm was developed for daily-ahead solar radiation forecasting in Chicago, USA. Two models were considered to benchmark the modelled algorithm. Performance metrics MAE (8.26 W/m^2) and MAPE (2.89%) show that the proposed model outperformed the benchmarked models [23]. However, the models developed in the latter two works tend to suffer premature convergence because of the nature of the PSO algorithm. The study in Ref. [24] combined coot algorithm (CA) with the RF algorithm to predict Australia's one-day solar radiation and wind speed. The prediction results show that the combined algorithm was superior to PSO-RF Local minimum convergence, which may impair the algorithm's performance.

One hour ahead of the solar radiation forecasting model, a comparative analysis was conducted using asynchronous dual-pipeline DL methods comprising MLP, LSTM,

CNN, and Transformer Networks [25]. Two Canadian weather data with cloud coverage were simulated and preprocessed using the min-max method. GridSearchCV optimization technique was used to modify the hyperparameters of the DL methods. The Friedman Statistical test results revealed that CNN (1.79) and MLP (2.25) performed best. A hybrid PSO-GA model was proposed to adjust the hyperparameters of ANN [26]. The resulting model was deployed to predict the hourly solar PV power output in Bhilai Chattisgarh, India. In Ref. [27], a comparative analysis of improved white shark optimization-based GRNN and RBFNN was conducted. The models were applied to predict 15 min and one hour ahead prediction of PV-wind RES in Malaysia and Turkey. EMD technique was utilized to decompose the weather and SCADA data. Regarding performance errors, the results showed that the improved RBFNN outperformed the improved GRNN and other benchmarked models.

In efforts to further improve the forecasting capability for solar radiation, some studies considered ridge regression for an ensemble of traditional ML algorithms. Carneiro et al. ensembled MLP, RBF, CFBP, and SOM ML algorithms with ridge regression to forecast one hour ahead of solar and wind RES in Brazil and Spain [28]. The ensemble model outperformed each traditional model with error values of RMSE (35.467), MAPE (14.191), and MAE (24.958). Likewise, ridge regression was utilized to ensemble extreme gradient boosting (XGB) and deep neural networks (DNN) to predict one-hour short-term solar radiation in three locations in India. The RMES (51.35) shows that the developed model is superior to the benchmarked algorithms [29]. Another ensemble model that comprises RF and CatBoost algorithms was developed using Bayesian Moving Average to forecast solar radiation and PV power production in Tripura, India [30]. Moreover, an amalgamation of CNN, RF, and LSTM using the min-max summation ensemble method was conducted to forecast 15 min and 24 h of solar radiation in Turin, Italy. A VMD approach was used to decompose the data, and the model's performance was compared with several algorithms [31]. The RMSE 83.8466 and MAE 45.4011 revealed the model's superiority over the benchmarked models. One obvious observation from these ensemble learning models is that they exhibit high-time computation compared to the benchmarked models.

It is evident from the review of the existing studies that the performance of any forecasting model is mainly based on model accuracy, performance errors, and computational time. The existing studies have, in one way or another, addressed some research gaps associated with solar radiation forecasting to the best of their capabilities. Nonetheless, it was discovered that model complexity and computation time increase as model accuracy increases due to the implementation of the hybrid and optimization techniques. Therefore, this justifies the "No Free Launch Theory" in ML algorithms. The main contributions of the work are

- Investigation of LSTM to forecast the GHR of the Desert Knowledge Australia Solar Centre (DKASC) Alice Spring meteorological decomposed data.
- Implementing adaptive hyperparameter adjustment strategies to develop a hybrid PSO-LSTM model.
- A comparative analysis of the developed hybrid models with three other metaheuristic algorithms to estimate the performance of each forecasting model for the solar renewable energy management system.

The remaining part of this article is organized: Sect. 2 gives a mini-review of the published articles, and Sect. 3 provides a brief theoretical background of the focused ensemble learning and DL algorithms. A full description of the materials and methods is provided in Sect. 4, while Sect. 5 presents the experimental results and discusses the analysis. Lastly, Sect. 6 caps this article up with a conclusion.

2 Theoretical Background of Forecasting Models

2.1 Long Short-Term Memory

There are now several ANN model variations, including CNN and DNN. These variants' input vectors are unrelated to one another. Distinct CNN and DNN, RNN creates sequential evidence from time series statistics, stores prior knowledge in the hidden state, and continuously updates fresh information. It can be used for prediction in various dynamic systems [32]. In other cases, the output of the RNN can be used as input to build a specified dynamic system. It has a unique, deep organization and works well in the time series functions. Nevertheless, RNN's limitations can be attributed to the intricate training techniques uncovered during tuning. This approach's typical disappearing or exploded gradient issues make learning the long-term dependencies challenging. To solve this RNN vulnerability, LSTM and Gated Recurrent Unit (GRU) were instigated [33]. They are variances of RNNs that can generate sequential information by learning long-term dependencies from time series statistics.

A memory cell, a hidden layer, in the LSTM piles information and repetitively determines new carries using gating mechanisms. The gates are the output, update, and forget gates [34]. The top element in the LSTM process is the cell state c . The input set is $x_k \in \mathcal{R}^n$, and the number of sets is n . Equation (1) represents the first gate, the forget gate, which controls the data deemed unnecessary from the cell state based on the previous process.

$$f_k = \sigma(W_f * [h_{k-1}, x_k] + b_f) \quad (1)$$

where σ symbolizes the sigmoid nonlinear activation function and b_f signifies the bias function of the forget gate.

The update gate p_k , which regulates the sum of data in the memory cell, is the second block of the LSTM. It has the candidate carry $\hat{c}$ and an input gate i_k . While the candidate carry, $\hat{c}$, indicates the candidate value that has to be updated in the memory state, the input gate i_k defines the memory state worth that should be updated. Equations (2), (3), and (4) give three components' mathematical expressions.

$$i_k = \sigma(W_i * [h_{k-1}, x_k] + b_i) \quad (2)$$

$$\hat{c}_k = \tanh(W_c * [h_{k-1}, x_k] + b_c) \quad (3)$$

$$p_k = i_k * \hat{c}_k \quad (4)$$

where b_i denotes bias of the input gate, b_c symbolized the bias of the candidate carry, and $\tanh$ signifies the hyperbolic tangent functions. Equation 5 gives the updated memory state, c_k .

$$c_k = p_k + f_k * c_{k-1} \quad (5)$$

LSTM's third gates include the output gate o_k and the output level h_k . The output gate o_k stays formulated as Eq. (6).

$$o_k = \sigma(W_o * [h_{k-1}, x_k] + b_o) \quad (6)$$

where b_o symbolizes the output gate bias function, o_k is used to adjust the selected function h_k , and it is extracted as Eq. 7.

$$h_k = o_k * \tanh(c_k) \quad (7)$$

LSTM's trainable weight vectors are W_f , W_i , W_c and W_o . Through recurrent back-propagation, the LSTM neural network implementation can handle the laborious learning process to store information for extended periods.

2.2 Particle Swarm Optimization

PSO was first presented by Kennedy and Eberhart in 1995 as a population-based optimization technique motivated by fish schooling patterns and bird flocking [35]. In general, PSO is a metaheuristic algorithm that belongs to the class of swarm intelligence. It is inspired by emerging intelligence and collective behaviour in a society that exhibits a well-arranged populace [36]. PSO is like other evolutionary computation methods in operation; it creates a population of random solutions to initialize the system and produce optimal solutions after successful iterations. PSO and evolutionary methods differ because PSO does not involve evolution operators in their procedures [15]. It develops the best solution to a given problem by utilizing a population of candidate solutions. The optimal intensity of every possible solution is calculated using a fitness function. The population's constituents are distributed throughout the problem's search space, and a particle's perspective can be used to explain the swarm's behaviour. A swarm is dispersed randomly throughout the examination space at the beginning of the procedure. The three components that influence a particle's motion are:

The three components that influence a particle's motion of the PSO algorithm are:

- i. Inertia component (w): this makes the particle trace its current path of movement.
- ii. Cognitive component (c_1): it enables the particle to move towards its own personal best.
- iii. Social component (c_2): this enables the particle to move towards the global best the whole swarm achieves.

The maneuver of the particle i connecting the iteration t and $t + 1$ is formulated as Eqs. 8 and 9.

$$v_i(t + 1) = wv_i(t) + c_1r_1(p_{bi}(t) - x_i(t)) + c_2r_2(g_b(t) - x_i(t)) \quad (8)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (9)$$

where $x_i(t)$ and $v_i(t)$ are the position and velocity of the particle i at the iteration t , respectively, p_{bi} represents the point of the best solution by which the particle has passed (personal best), and g_b denotes the locus of the best-attained solution of the complete swarm (global best). c_1 and c_2 are two numerical values that symbolize the acceleration multipliers. They can be altered in some instances, contingent on the optimization problem. r_1 and r_2 are two random numbers from space $[0, 1]$. The PSO procedure begins with randomly initializing the particles in their search space, appointing the particles a preliminary position and velocity. The particles move based on Eqs. 1 and 2, and the fitness (objective) function is evaluated at each iteration of the algorithm to calculate the particles' best positions p_{bi} and global best g_b of the whole swarm. The procedure keeps running and updating until the stopping criteria are satisfied. Like most optimization methods, PSO performs exploration and exploitation. The most favourable regions in the search space are identified through exploration, while the convergence of the particles toward the best solution is handled through exploitation. PSO algorithm consistently exhibits quicker convergence; it can handle nonlinear, nondifferential objective functions and an extensive search space dataset [37].

3 Development of Solar Radiation Models: Materials and Methods

3.1 Data Source and Preprocessing

Renewable energy prediction analysis requires effective and efficient data gathering and preprocessing methods. This study will explore an existing dataset from DKASC, Alice Spring database [38]. This solar center is one of Australia's largest tilted and diffused PV panel systems. Figure 1 represents the study area location in Australia. It comprises modern, sophisticated equipment for acquiring meteorological datasets. Many data features were recorded during the data capture. However, the features considered for this study are GHR, temperature, pressure, relative humidity, wind direction, and wind speed. The original data was documented in 5-min spaces daily from September 29, 2016, to December 1, 2016. The entirety number of instances documented in the period was 32,686. The dataset's missing values were replaced with each feature's mean value. This study uses all the recorded instances of the DKASC Alice Spring dataset because there is no record of zero value for GHR.

Figure 2 presents the seasonal variations of GHR in two-dimensional surface plots of the data set for seven days in September. The GHR is low in the morning and then increases gradually until it reaches its maximum value midday. As the day falls, the GHR's value decreases gradually until it reaches its lowest value. This phenomenon repeats itself daily, which is termed seasonal variation. During the summer, the value of GHR is higher and becomes lower during the winter. However, observing these seasonal variations cannot help us understand many hidden patterns. It requires the application of emerging technologies. Figure 3 illustrates the implementation of the emerging technologies deployed in this study. Moreover, studies have shown that the power productivity of RES, especially solar and wind, depends on the variability of meteorological features [3].

These features have different correlation coefficients on one another. This study utilizes Pearson correlation, a statistical technique determining the linear correlation between the meteorological features. The technique takes the coefficient values between $+1$ and -1 ; $+1$ signifies a seamless positive linear correlation, while -1 specifies a seamless negative linear correlation [39]. Figure 4 illustrates a heatmap that presents the linear correlations between the GHR and other meteorological features (temperature, pressure, relative humidity, wind direction, and wind speed). The heatmap reveals a positive relationship ($+0.73$) between GHR and temperature, ($+0.12$) between GHR and pressure, and ($+0.073$) between GHR and wind speed. This reveals that the higher the temperature, pressure, and wind speed are, the higher the GHR is. Nonetheless, relative humidity and wind direction have the same negative linear correlation (-0.23) with GHR, indicating that GHR decreases as relative humidity and wind speed increase. This represents an adverse effect on the GHR. Pearson correlation analysis focuses on linear relationships between the GHR and meteorological variables; some patterns require AI techniques.

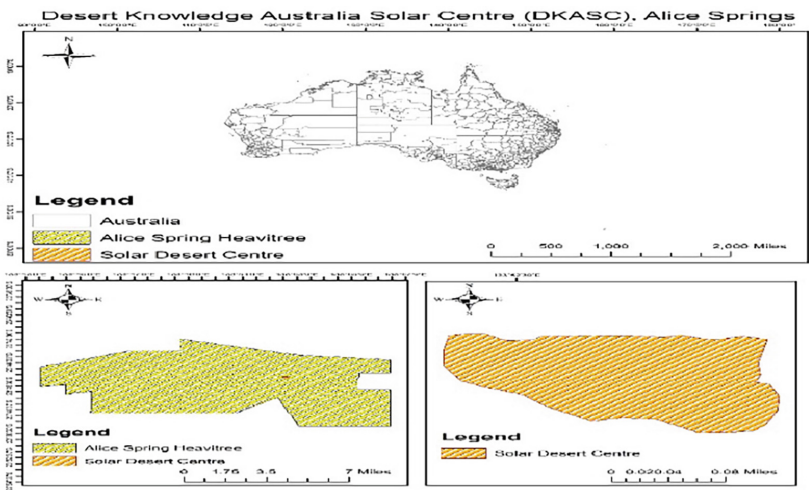


Fig. 1. Desert Knowledge Australia Solar Center (DKASC), Alice Spring – Study Area

3.2 Forecasting Model Using Hybrid PSO-LSTM Model

The deep learning algorithm, especially LSTM, exhibits formidable abilities to forecast time series models. The LSTM is utilized in this study because it can model long-term dependencies regarding time series datasets. LSTM hyperparameters were adaptively adjusted with the four metaheuristic algorithms that resulted in PSO-LSTM, GA-LSTM, GWO-LSTM, and ACO-LSTM solar radiation forecasting models. The hyperparameters adjusted by the metaheuristic algorithms are LSTM unit and dropout rate, and the bound of these two hyperparameters are 16, 128, and 0.1, 0.5, respectively. The best values of these two bounds were used to train the LSTM models. The summary of the

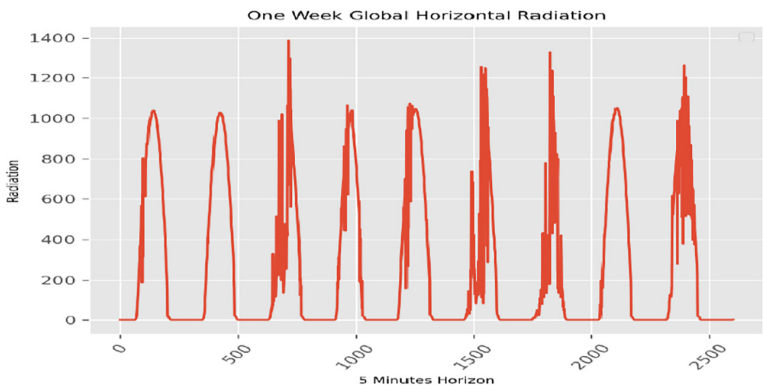


Fig. 2. A seasonal variability of global horizontal radiation in 5-min horizons.

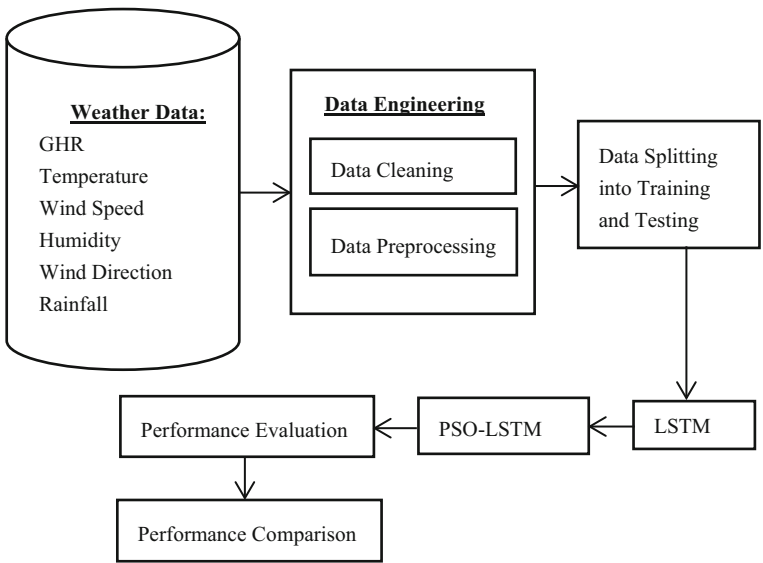


Fig. 3. A schematic of a PSO-LSTM forecasting model for global horizontal radiation.

hyperparameters considered is presented in Table 1. Adam optimizer was selected due to its adaptive learning rate and ability to enhance performance accuracy. A mean squared error was chosen as an objective function that minimized the error between the targets and predicted outcomes. Likewise, the parameters of the metaheuristic algorithms and their values are presented in Tables 2 and 3.

3.3 Performance Evaluation

All the models developed for forecasting solar radiation were evaluated using four performance metrics. The metrics are:

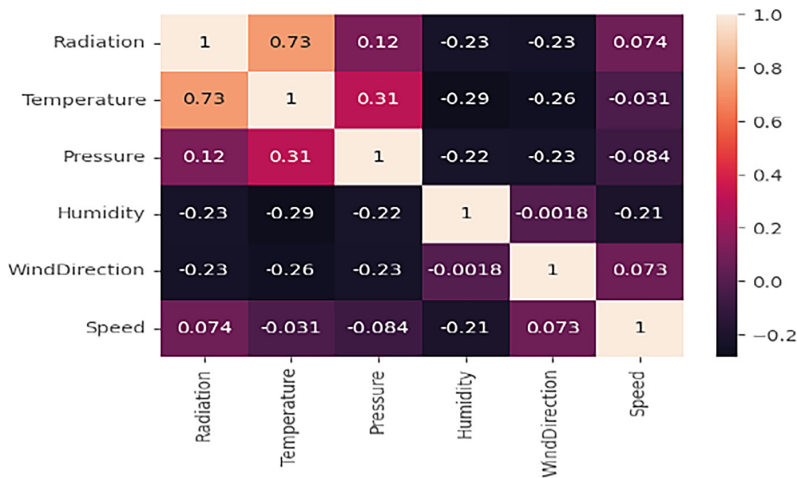


Fig. 4. The heatmap depicting the correlation coefficient of the features considered from DKA Alice Spring 2015 datasets.

Table 1. Hyperparameters for the development of LSTM models.

Hyperparameters	Value
Number of layers	14, 6
LSTM unit	Bound (16, 128)
Activation function	Sequential
Epoch	10
Batch size	32
Dropout	Bound (0.1, 0.5)
Optimizer	Adam
Loss function	Mean squared error

i. R Score (R^2) was used to measure the accuracy of the developed solar radiation forecasting models; its mathematical expression is given in Eq. 10. The larger the percentage of R^2 , the better the model's performance.

$$R^2 = \frac{\sum_{i=1}^n (\hat{y}_n - \bar{y})^2}{\sum_{i=1}^n (y_n - \bar{y})^2} \tag{10}$$

ii. Relative Mean Square Error (RMSE) determines the difference between the measured and trained data. Equation 11 provides the RMSE mathematical expression. The smaller the RMSE, the better the solar radiation forecasting model.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2} \tag{11}$$

Table 2. The PSO and GA parameters for LSTM optimization

PSO		GA	
Parameter	Value	Parameter	Value
Inertia weight (w)	0.5	Population size	100
Cognitive constant (c_1)	0.5	Mutation probability	0.1
Social constant (c_2)	0.5	Crossover probability	0.5
Swarm size	100	Elite ratio	0.01
Maximum iteration	100	Parents portion	0.3
		Crossover type	Uniform

Table 3. The GWO and ACO parameters for LSTM optimization

GWO		ACO	
Parameter	Value	Parameter	Value
Pack size	5	Colony size	25
Number of Iteration	50	Pheromone exponential factor	1.0
Maximum value	100	Heuristic information exponential factor	2.0
Minimum value	−100	Pheromone evaporation coefficient	0.1
		Best path probability	0.9

- iii. Mean Absolute Error (MAE): This metric takes the absolute difference between the trained and measured data divided by the number of instances. It is formulated as Eq. 12.

$$MAE = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i| \quad (12)$$

- iv. Mean Absolute Percentage Error (MAPE): this is formulated by taking the percentage of MAE as expressed in Eq. 13. It is commonly used to determine the accuracy of the time series model.

$$MAPE = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i| \times 100 \quad (13)$$

where n , $\hat{y}_i$, and y_i are the total number of data instances, radiation predicted, and actual value, respectively.

4 Results and Discussion

The effective forecasting of RES is required for an effective renewable energy management system. Seasonal variabilities of meteorological features impact solar radiation, which impacts power out of solar power systems. For this reason, solar radiation

requires effective forecasting mechanisms. This study developed AI models capable of self-adaptive learning of hidden patterns and accurately forecasting global horizontal radiation. DKASC Alice Spring datasets were explored; 70% were dedicated to training the models, while the remaining 30% were used for testing the models. A hybrid PSO-LSTM model was developed and compared with the other four LSTM-based models. Moreover, the models were evaluated with four performance metrics, namely, R^2 , RMSE, MAE, and MAPE. The total number of models developed to forecast the GHR is twenty models. This section presents the experimental results and discusses the models that were developed. The experiment was conducted on the Python platform using Numpy, Pandas, SKLearn, Keras, and TensorFlow packages. Also, geneticalgorithm, pyswarm, and pyMetaheuristic libraries were explored for GA, PSO, and GWO implementation, respectively. The libraries for ACO were developed using the same Python platform. The system information for this study is an x64-based PC with Processor - Intel(R) Core (TM) i7-4600U CPU 2.10 GHz, 2701 MHz, 2 Core(s), and 4 Logical Processor(s).

This study developed a hybrid POS-LSTM model for the short-term forecasting of GHR using the region of DKASC Alice Spring in Australia. Thorough analysis and comparison of this model were conducted with traditional LSTM, GA, GWO, and ACO-based models. All the models performed to their best capabilities in forecasting the GHR of the region, as shown in Table 4. However, the performance evaluations of the GWO and ACO-based models were below those of the traditional LSTM model. This revealed that optimizing renewable energy forecasting models with GWO and ACO algorithms might require extensive improvement of the two algorithms, which might also lead to high model complexity.

Table 4. Comparison of PSO-LSTM and benchmarked models performance evaluations.

Model		R^2	RMSE	MAE	MAPE
LSTM	Training	0.936912	0.251214	0.115424	227.524346
	Testing	0.943503	64.519835	30.153927	287.554496
GA - LSTM	Training	0.942175	0.240507	0.100509	237.362980
	Testing	0.948598	61.541874	27.541454	309.640835
GWO - LSTM	Training	0.942784	0.239239	0.099615	237.580819
	Testing	0.941280	65.776560	28.793149	239.924262
ACO - LSTM	Training	0.942437	0.239963	0.100249	236.190306
	Testing	0.929778	71.930958	30.312709	228.627622
Our Model: PSO - LSTM	Training	0.942837	0.239128	0.100781	235.298517
	Testing	0.950643	60.304873	25.442901	189.470553

Moreover, this study underscores the LSTM model and its variants' superiorities forecasting capabilities and discusses their outstanding performances in improving renewable management systems. Figure 5 presents the loss per epoch of LSTM and its optimized variants. It can be observed that all the models experience decay in their errors.

Also, Fig. 6 depicts the forecasts versus actual measurements comparisons of the LSTM and its variants. Figure 5 and 6 showcase the potential of the LSTM model and its adjustment of self-adaptive parameters through implemented metaheuristic algorithms. The R^2 results showed that the superiority of LSTM and its optimized models can be credited to its inherent architecture. The models can escape the vanishing gradient, which resourcefully captures the long-term temporal dependencies in the highly variable and nonlinearity of GHR and meteorological data. These capabilities of the LSTM models encourage their acceptance of handling energy management in the renewable energy domain.

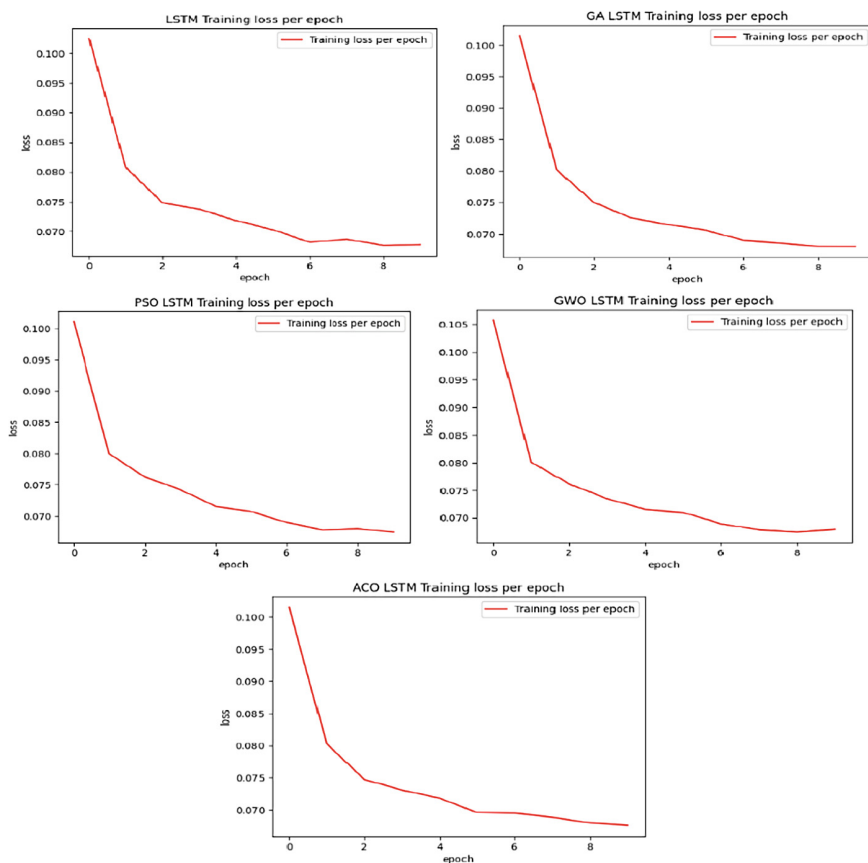


Fig. 5. Comparison of loss per epoch of the variances of LSTM models developed.

Furthermore, GA-LSTM outperformed the LSTM, GWO-LSTM, and ACO-LSTM models with 0.943503 and 0.942175 R^2 score values for training and testing, respectively. This outstanding performance of the GA-LSTM model might be due to its capability to maintain population diversity in evolution. The proposed hybrid model, PSO-LSTM, is the most superior. The combination of the potential features of PSO, such as the

ability to perform exploration and exploitation, handling of nonlinear problems, quicker convergence, and its simplicity with the strength of the LSTM model, led to its best performance. The performance evaluations of the experimental result revealed that the metaheuristic algorithms could handle the shortcomings of the traditional LSTM model for the forecasting of GHR. Solar energy power production significantly depends on solar radiation and other meteorological features. The features exhibit intermittency that impairs solar energy output. Integration of PSO-based models, as revealed in this study, PSO-LSTM with accuracy (0.950643), will better forecast the GHR, which invariably helps the accurate forecast of solar energy production. Additionally, accurate forecasts of GHR encourage effective integration of solar energy systems into the microgrid and smart grid systems. Implementing PSO-LSTM models will help the stakeholders and policymakers make informed decisions about the renewable energy system.

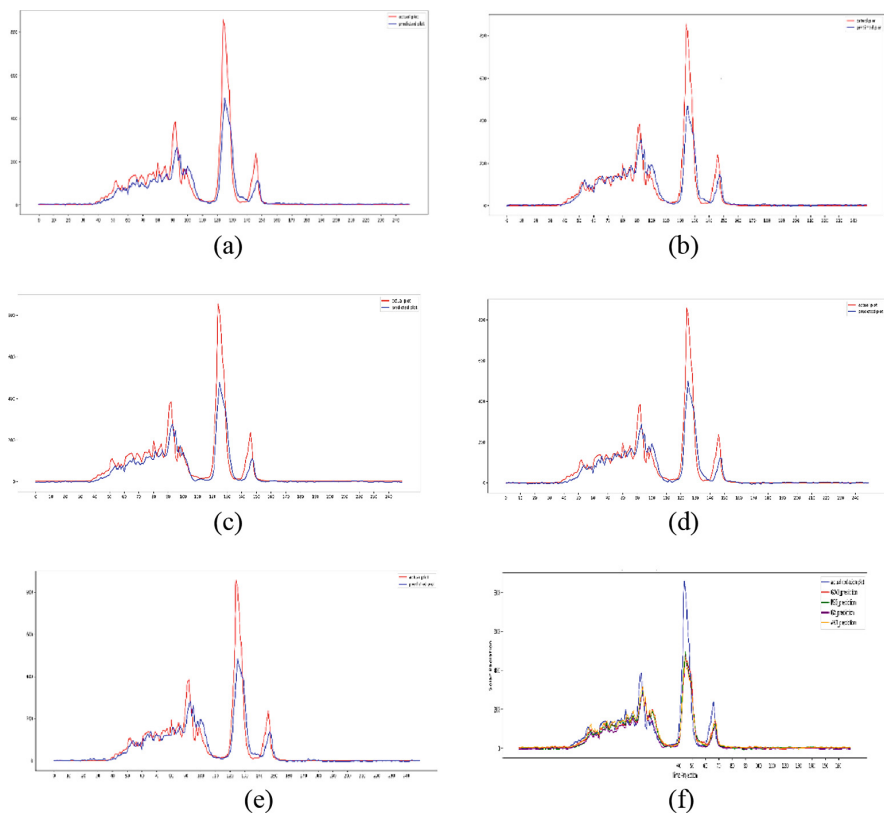


Fig. 6. Comparison of (a) LSTM forecasting output versus actual measurements, (b) ACO-LSTM forecasting output versus actual measurements, (c) GA-LSTM forecasting output versus actual measurements, (d) PSO-LSTM forecasting output versus actual measurements, (e) GWO-LSTM forecasting output versus actual measurements, and (f) all the GA-LSTM, PSO-LSTM, GWO-LSTM, ACO-LSTM, and the actual measurements

In conclusion, this study showcased the potential of metaheuristic algorithms in boosting the performance of the LSTM model. Also, it was revealed that PSO-LSTM has the best performance evaluations. Notwithstanding, the PSO algorithm requires more improvements, such as improving the learning strategy and rate of the particles to balance the algorithm's exploration and exploitation strategy. Doing this will strengthen the PSO algorithm to overcome the shortcomings of local optima and premature convergence. Achieving this improvement, which will serve as future work, will strengthen the capability of the ensemble and deep learning models considered in this study.

5 Conclusion

Several studies have shown that RES has excellent potential to overcome the challenges of using fossil fuels in the electric power generation industry. As the most affordable RES, the solar system depends most on solar radiation and other meteorological parameters. Solar radiation and these meteorological parameters are intermittent and affect solar energy output. For these reasons, this study proposed a hybrid PSO-LSTM model to forecast the GHR of the region of Alice Springs in Australia. The PSO algorithm was implemented to conduct self-adaptive parameter adjustment of the traditional LSTM models. Three other metaheuristic algorithms (GA, GWO, and ACO) were utilized to compare the proposed model. Four evaluation metrics (R^2 , RMSE, MAE, and MAPE) were used to measure all the developed models' performances. The results revealed that all the models could forecast GHR to the best of their capabilities. The outcomes of PSO-LSTM with R^2 (0.950643), RMSE (60.304873), and MAE (25.442901) revealed that the proposed model is superior to the traditional LSTM and other optimized models. The ability of the PSO algorithm to perform exploration and exploitation over a specified search space and the strength of the LSTM model to handle the vanishing gradient promoted the GHR forecasting accuracy of the PSO-LSTM model. Nonetheless, there are several strategies for modifying the PSO algorithm to improve the performance of the PSO-LSTM. These strategies will help the PSO-LSTM model to overcome the shortcomings of local optimal and premature convergence. Therefore, these strategies will serve as our future directions. The findings in this study underscore the capabilities of AI-based models in the RES domain. The outcomes of the developed techniques improved the accuracy of the forecasting of the GHR. Studying and implementing these models in the RES domain will help researchers, policymakers, and stakeholders revolutionize electric energy production and bring sustainable development.

Acknowledgments. This work was partly supported by the South African National Research Foundation under Grant nos. CPRR240327211161, CHN231229201835 and AJCR230704126719120106.

Author Contribution. **Blessing Olatunde Abisoye:** Conceptualization, Resources, Fund acquisition, Methodology, Writing – original draft, software, Data curation, Visualization, Formal analysis. **Yanxia Sun:** Conceptualization, Resources, Fund acquisition, Supervision, Validation, Writing - review and editing, Project administration. **Wang Zenghui:** Validation, Resources, Fund Acquisition, Investigation, Writing - review and editing.

Data Availability. DKASC, “Alice Springs of Desert Knowledge Australia Centre.” Accessed: Apr. 20, 2023. [Online]. Available: <http://dkasolarcentre.com.au/historical-data/download>.

Declaration of Competing Interest. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Cabello-López, T., Carranza-García, M., Riquelme, J.C., García-Gutiérrez, J.: Forecasting solar energy production in Spain: a comparison of univariate and multivariate models at the national level. *Appl. Energy* **350** (2023). <https://doi.org/10.1016/j.apenergy.2023.121645>
2. Sorkun, M.C., Incel, Ö.D., Paoli, C.: Time series forecasting on multivariate solar radiation data using deep learning (LSTM). *Turkish J. Electr. Eng. Comp. Sci.* **28**(1), 211–223 (2020). <https://doi.org/10.3906/elk-1907-218>
3. Abisoye, B.O., Sun, Y., Zenghui, W.: A survey of artificial intelligence methods for renewable energy forecasting: methodologies and insights. *Renew. Energy Focus* **48**, 100529 (2024). <https://doi.org/10.1016/j.ref.2023.100529>
4. Belmahdi, B., Madhiarasan, M., Louzazni, M., El Bouardi, A.: Forecasting solar radiation using machine learning method: new optimization algorithm. In: Moldovan, L., Gligor, A. (eds.) *The 16th International Conference Interdisciplinarity in Engineering: Inter-Eng 2022 Conference Proceedings*, pp. 706–717. Springer International Publishing, Cham (2023). https://doi.org/10.1007/978-3-031-22375-4_56
5. Yadav, A.K., Chandel, S.S.: *Solar Radiation Prediction Using Artificial Neural Network Techniques: A Review*. Elsevier Ltd. (2014). <https://doi.org/10.1016/j.rser.2013.08.055>
6. Ghimire, S., Deo, R.C., Wang, H., Al-Musaylh, M.S., Casillas-Pérez, D., Salcedo-Sanz, S.: Stacked LSTM sequence-to-sequence autoencoder with feature selection for daily solar radiation prediction: a review and new modeling results. *Energies (Basel)* **15**(3) (2022). <https://doi.org/10.3390/en15031061>
7. Malik, P., Gehlot, A., Singh, R., Gupta, L.R., Thakur, A.K.: A review on ANN based model for solar radiation and wind speed prediction with real-time data. *Arch. Comp. Meth. Eng.* **29**(5), 3183–3201 (2022). <https://doi.org/10.1007/s11831-021-09687-3>
8. Quej, V.H., Almorox, J., Arnaldo, J.A., Saito, L.: ANFIS, SVM and ANN soft-computing techniques to estimate daily global solar radiation in a warm sub-humid environment. *J. Atmos. Sol. Terr. Phys.* **155**, 62–70 (2017). <https://doi.org/10.1016/j.jastp.2017.02.002>
9. Ghimire, S., Bhandari, B., Casillas-Pérez, D., Deo, R.C., Salcedo-Sanz, S.: Hybrid deep CNN-SVR algorithm for solar radiation prediction problems in Queensland, Australia. *Eng. Appl. Artif. Intell.* **112** (2022). <https://doi.org/10.1016/j.engappai.2022.104860>
10. Abdallah, M., Mohammadi, B., Nasiri, H., Katipoğlu, O.M., Abdalla, M.A.A., Ebadzadeh, M.M.: Daily global solar radiation time series prediction using variational mode decomposition combined with multi-functional recurrent fuzzy neural network and quantile regression forests algorithm. *Energy Rep.* **10**, 4198–4217 (2023). <https://doi.org/10.1016/j.egyr.2023.10.070>
11. Abisoye, B.O., Sun, Y., Wang, Z.: Machine learning forecasting model for solar energy radiation. *Int. J. Comp. Theor. Eng.* **16**(2), 66–75 (2024). <https://doi.org/10.7763/IJCTE.2024.V16.1355>
12. Villegas-Mier, C.G., Rodriguez-Resendiz, J., et al.: Optimized random forest for solar radiation prediction using sunshine hours. *Micromachines (Basel)* **13**(9) (2022). <https://doi.org/10.3390/mi13091406>

13. Kumar, M., Namrata, K., Kumar, N.: Data-driven hyperparameter optimized extreme gradient boosting machine learning model for solar radiation forecasting. *Adv. Electr. Electron. Eng.* **20**(4), 549–559 (2023). <https://doi.org/10.15598/aeec.v20i4.4650>
14. VanDeventer, W., et al.: Short-term PV power forecasting using hybrid GASVM technique. *Renew. Energy* **140**, 367–379 (2019). <https://doi.org/10.1016/j.renene.2019.02.087>
15. Adaryani, F.R., Jamshid Mousavi, S., Jafari, F.: Short-term rainfall forecasting using machine learning-based approaches of PSO-SVR, LSTM and CNN. *J. Hydrol. (Amst.)* **614** (2022). <https://doi.org/10.1016/j.jhydrol.2022.128463>
16. Moazen-zadeh, R., Mohammadi, B., Duan, Z., Delghandi, M.: Improving generalisation capability of artificial intelligence-based solar radiation estimator models using a bio-inspired optimisation algorithm and multi-model approach. *Environ. Sci. Pollut. Res.* **29**(19), 27719–27737 (2022). <https://doi.org/10.1007/s11356-021-17852-1>
17. Akhter, M.N., et al.: A hybrid deep learning method for an hour ahead power output forecasting of three different photovoltaic systems. *Appl. Energy* **307**, (2022). <https://doi.org/10.1016/j.apenergy.2021.118185>
18. Tawn, R., Browell, J.: A review of very short-term wind and solar power forecasting. *Renew. Sustain. Energy Rev.* **153**, 1–15 (2021). <https://doi.org/10.5281/zenodo.5070758>
19. Ghimire, S., Deo, R.C., Casillas-Pérez, D., Salcedo-Sanz, S.: Improved complete ensemble empirical mode decomposition with adaptive noise deep residual model for short-term multi-step solar radiation prediction. *Renew. Energy* **190**, 408–424 (2022). <https://doi.org/10.1016/j.renene.2022.03.120>
20. Neshat, M., Nezhad, M.M., Mirjalili, S., Garcia, D.A., et al.: Short-term solar radiation forecasting using hybrid deep residual learning and gated LSTM recurrent network with differential covariance matrix adaptation evolution strategy. *Energy* **278** (2023). <https://doi.org/10.1016/j.energy.2023.127701>
21. Wang, Y., et al.: Hybrid solar radiation forecasting model with temporal convolutional network using data decomposition and improved artificial ecosystem-based optimization algorithm. *Energy* **280** (2023). <https://doi.org/10.1016/j.energy.2023.128171>
22. Sunayana, G., Anudeep, R.K., Yashi, B., Rajesh, K.: Hybrid random forest and particle swarm optimization algorithm for solar radiation prediction. In: *IEEE 5th International Conference on Computing Communication and Automation (ICCCA)*, pp. 302–307 (2020)
23. Sreenu, S., Kailash, C.S., Rohit, B.: Optimized support vector regression models for short term solar radiation forecasting in smart environment. In: *2016 IEEE Region 10 Conference (TENCON)*, pp. 1929–1932 (2016)
24. Islam, M.M., Nagrial, M., Rizk, J., et al.: Solar radiation and wind speed forecasting using deep learning technique. In: *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering, CSDE 2021*. Institute of Electrical and Electronics Engineers Inc. (2021). <https://doi.org/10.1109/CSDE53843.2021.9718372>
25. Lara-Benítez, P., Carranza-García, M., Luna-Romera, J.M., Riquelme, J.C.: Short-term solar irradiance forecasting in streaming with deep learning. *Neurocomputing* **546** (2023). <https://doi.org/10.1016/j.neucom.2023.126312>
26. Roy, R., Varghese, A.J., Awasthi, S.R.: Compare and evaluate one hour ahead solar power forecasting with ANN tuned using genetic algorithm and hybrid genetics-based particle swarm optimization. In: *Lecture Notes in Electrical Engineering*, pp. 405–417. Springer Science and Business Media Deutschland GmbH (2022). https://doi.org/10.1007/978-981-19-0193-5_33
27. Mansoor, M., Feroz Mirza, A., Usman, M., et al.: Hybrid forecasting models for wind-PV systems in diverse geographical locations: Performance and power potential analysis. *Energy Convers. Manag.* **287** (2023). <https://doi.org/10.1016/j.enconman.2023.117080>
28. Carneiro, T.C., Rocha, P.A.C., Carvalho, P.C.M., et al.: Ridge regression ensemble of machine learning models applied to solar and wind forecasting in Brazil and Spain. *Appl. Energy* **314** (2022). <https://doi.org/10.1016/j.apenergy.2022.118936>

29. Kumari, P., Toshniwal, D.: Extreme gradient boosting and deep neural network based ensemble learning approach to forecast hourly solar irradiance. *J. Clean Prod.* **279** (2021). <https://doi.org/10.1016/j.jclepro.2020.123285>
30. Banik, R., Biswas, A.: Improving solar PV prediction performance with RF-CatBoost ensemble: a robust and complementary approach. *Renew. Energy Focus* **46**, 207–221 (2023). <https://doi.org/10.1016/j.ref.2023.06.009>
31. Cannizzaro, D., Aliberti, A., Bottaccioli, L., et al.: Solar radiation forecasting based on convolutional neural network and ensemble learning. *Exp. Syst. Appl.* **181** (2021). <https://doi.org/10.1016/j.eswa.2021.115167>
32. Jiang, M., Liu, J., Zhang, L., Liu, C.: An improved Stacking framework for stock index prediction by leveraging tree-based ensemble models and deep learning algorithms. *Phys. A Statist. Mech. Appl.* **541** (2020). <https://doi.org/10.1016/j.physa.2019.122272>
33. Zarzycki, K., Ławryńczuk, M.: LSTM and GRU neural networks as models of dynamical processes used in predictive control: A comparison of models developed for two chemical reactors. *Sensors* **21**(16), 1–27 (2021). <https://doi.org/10.3390/s21165625>
34. Guimarães Da Silva, D., et al.: Performance evaluation of LSTM neural networks for consumption prediction. *Electron. Energy* **2**, 1–14 (2022). <https://doi.org/10.1016/j.prime.2022.100030>
35. Kennedy, J., Eberhart, R.: Particle Swarm Optimization, pp. 1942–1948. IEEE (1995)
36. Ali Jallal, M., Chabaa, S., Zeroual, A.: A novel deep neural network based on randomly occurring distributed delayed PSO algorithm for monitoring the energy produced by four dual-axis solar trackers. *Renew. Energy* **149**, 1182–1196 (2020). <https://doi.org/10.1016/j.renene.2019.10.117>
37. Behera, M.K., Majumder, I., Nayak, N.: Solar photovoltaic power forecasting using optimized modified extreme learning machine technique. *Eng. Sci. Technol. Int. J.* **21**(3), 428–438 (2018). <https://doi.org/10.1016/j.jestech.2018.04.013>
38. DKASC: Alice Spring of Desert Knowledge Australia Centre. <http://dkasolarcentre.com.au/historical-data/download>. Accessed 20 April 2023
39. Unsal, D.B., Aksoz, A., Oyucu, S., Guerrero, J.M., Guler, M.: A comparative study of AI methods on renewable energy prediction for smart grids: case of turkey. *Sustainability (Switzerland)* **16**(7) (2024). <https://doi.org/10.3390/su16072894>



Modelling Road Networks with Node Structural Features and Graph Convolutional Networks

Mohale Molefe^(✉) 

University of KwaZulu Natal, Durban, South Africa
molefemohale@gmail.com

Abstract. Machine learning-based classification of roads is crucial for the effectiveness of smart transportation infrastructure. Recent road-type classification methods rely on road attribute features obtained from the publicly available OpenStreetMaps platform. However, the absence of certain road attributes often makes it difficult to develop models for monitoring road networks using OpenStreetMaps. Presented in this study is an approach for classifying roads where features are extracted using node structural feature generation methods and Graph Convolution Networks; without relying on road attribute features. The proposed method was evaluated on road network graph datasets of realistic cities. Experiments demonstrate that the presented method is capable of effectively modelling road networks even with missing road attributes.

The results indicate that the proposed method can model road networks with missing road attributes.

Keywords: road networks · road attribute features · node structural features · graph convolution networks · OpenStreetMap

1 Introduction

Accurate representation of road network maps is essential for designing and developing Intelligent Road Network Systems (IRNS). The ultimate objective of IRNS is to minimise traffic-related problems by providing road users with prior traffic information. IRNS design predominantly relies on Internet of Things (IoT) enabled devices (e.g., sensors, and cameras) that continuously monitor traffic congestion, volumes, and road conditions across the infrastructure. Access to the data captured by IoT devices has enabled the development of various machine learning-based IRNS models, including estimated time of arrival (ETA) and traffic flow predictions [1–3]. A common use of machine learning in IRNS is the development of road-type classification models; these models aim to classify road segments into road-types such as highways, links and residential roads. Therefore, road-type classification models are becoming important as they can be integrated into inter-activate maps to provide important traffic information to road users.

Recent road-type classification models [4–6] are built from road network data extracted from the publicly available OpenStreetMaps (OSM) mapping platform. Designed as an alternative to commercial ones, OSM is a crowd-sourced, freely editable mapping platform that contains some of the most detailed road information of realistic cities worldwide. However, OSM often has missing road segment attributes and inaccurate tags, as volunteers with different mapping skills mainly edit it. Despite its incompleteness and quality issues, OSM remains the widely used mapping platform for modelling road networks, and many attempts have been made to improve its quality [7–9]. A suitable alternative for OSM is a high-resolution remote sensing image dataset that often has complete ground coverage and is up to date with no missing information. However, the remote-sensing image dataset is often obscured by trees, vehicles, and buildings. This noisy image background information makes modelling road networks with remote sensing image data challenging. Therefore, road network extraction with sophisticated deep-learning approaches is mandatory to extract road networks from remote sensing image datasets.

In this work, a new feature representation method for modelling road networks is proposed. Contrary to the road-type classification method found in the literature, where road segment attributes are used to generate road segment features, this work takes full advantage of the segment’s spatial connection in a network to represent roads using structural features, with no reliance on road segment attributes.

The remainder of this paper is organised as follows: Sect. 2 provides a comprehensive review of recent advancements in machine learning for classifying roads. Section 3 details the methodologies and models introduced in this work. Section 4 presents the experimental findings and their analysis. Finally, Sect. 5 offers a conclusion and outlines directions for future research.

2 Related Work

The spatial and topological configuration of a road network is typically represented as a graph comprising nodes and edges. In this representation, nodes correspond to critical points in the network, such as dead-ends, junctions, and points of interest, while edges denote the road segments interconnecting these nodes. Unlike traditional, Euclidean based types of data (text and images), the representation of road networks using deep learning techniques poses a challenge due to the non-Euclidean and dynamic nature of graph-structured data. As a result, various graph representation learning (GRL) methods have been proposed in the literature to effectively model such graphs.

The main objective of GRL is to construct an embedding function that takes high-dimensional features in a non-Euclidean graph space and produces low-dimensional features in an Euclidean space. Some of the commonly known GRL methods include Graph Convolution Networks (GCN) [10], Graph SAmple and aggreGatE (GraphSAGE) [11] and Graph Attention Networks (GAT) [12]. GCN obtains an embedded vector of a target node by aggregating features from all

its connected neighbours for a given number of hops. GraphSAGE obtains an embedded vector of a target node by aggregating features only from a sampled number of its neighbours. GAT uses the attention mechanism to weigh the importance of each neighbour towards the target node.

Numerous approaches have been introduced in the literature for classifying roads utilising GRL techniques. In [4], a method for classifying distinct road types on a road network graph dataset sourced from OpenStreetMaps (OSM) is proposed. The authors initially converted the road network graph into a line graph, where road segments are represented as nodes, to enhance the applicability of GRL methods. Subsequently, feature vectors for each road segment were constructed using segment length, width, speed limit, and geometry. A comparative analysis of various GRL techniques was conducted to evaluate their effectiveness in modeling road-type classification across different learning scenarios.

In [5], the road-type classification problem is modelled using Deep AutoEncoder (DAE) as the embedding method. Similar to the work presented in [4], road segment attributes were used to generate feature vectors, however, DAE was proposed to embed the dimensionality of the original features while preserving the significant ones. Road-type classification on the embedded vectors was then achieved using a Multi-Layer Perception (MLP) classifier. The study presented in [6] proposed a multi-stage GRL method where features obtained by the DAE method proposed in [5] are fed as inputs to the GCN method before classifying road-types with the MLP classifier.

Despite achieving very effective results, the road-type classification approaches presented in the literature lack scalability as they rely on the road segment attributes obtained from OSM to represent each road segment. Even though the topological structure of any road network can be obtained from OSM, road attributes are often missing and inaccurate. Thus, a model that performs well in one road network graph might not achieve similar performance in another despite both networks being from the same region. In this work, feature representation using road segment attributes is replaced with feature representation using the topological structure of the road network. By leveraging how each road segment connects to other road segments in a network, more robust features can be obtained.

3 Materials and Methods

Depicted in Fig. 1 is the methodological diagram of the proposed method. Johannesburg's road network graph dataset is extracted from OpenStreetMaps (OSM). Thereafter, line graph transformation is applied to represent road segments as nodes. Then, road segment feature representation is generated by characterizing the structure and position of each road segment in a network. Graph Convolution Network (GCN) is applied to the generated features to represent a vector of each road segment by aggregating information from its neighbours. Finally, road-type classification is achieved using the MultiLayer Perceptron (MLP) classifier.

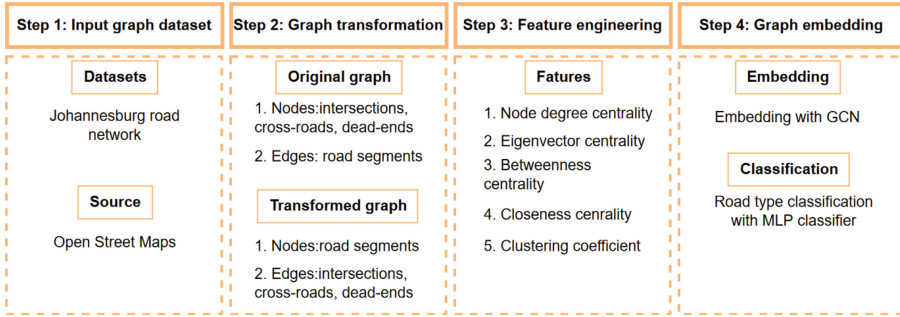


Fig. 1. Methodological flow diagram of the presented approach

3.1 Raod Network Graph Transformation

The input road network graph dataset for Johannesburg, derived from OSM, is denoted as $G = (V, E)$, where V and E represent the set of nodes and edges, respectively. The nodes correspond to intersections, junctions, and dead-ends, while the edges represent the road segments connecting these nodes. To facilitate road network modeling with GCN, the standard line graph transformation method [4–6] is applied. In this transformation, the nodes of the new graph $L(G)$ are the road segments, and an edge is established between two road segments in $L(G)$ if they share a common node in the original graph G .

3.2 Feature Engineering

Contrary to the feature engineering methods found in the literature where road segment attributes (e.g. road segment width, length, lane count) are used to generate its vector representation, this work uses node structural methods for feature representation. Structural features characterise the structure and position of the node in the network by describing its topological patterns. Two standard methods are used: centrality-based features and structure-based features. To explain how each method was applied to the network, a reference to the road network sub-graph and its corresponding adjacency matrix as shown in Figs. 1 and 2, respectively, will be made.

Centrality Based Methods. Centrality-based methods are a measure of how important a node is in the network. Centrality-based methods used in this work are discussed below.

A) Node Degree Centrality: Node degree centrality quantifies the significance of a node based on the number of neighboring nodes it is connected to. A higher degree indicates greater importance of the node within the network. As illustrated in Fig. 4, road segment B is connected to three neighboring road segments, thus its node centrality value is 3. The normalised node degree centrality

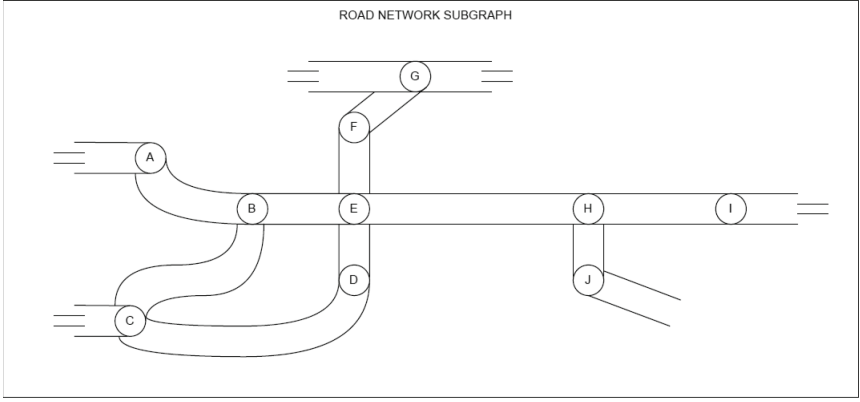


Fig. 2. Road network sub-graph

	A	B	C	D	E	F	G	H	I	J
A	0	1	0	0	0	0	0	0	0	0
B	1	0	1	0	1	0	0	0	0	0
C	0	1	0	1	0	0	0	0	0	0
D	0	0	1	0	1	0	0	0	0	0
E	0	1	0	1	0	1	0	1	0	0
F	0	0	0	0	1	0	1	0	0	0
G	0	0	0	0	0	1	0	0	0	0
H	0	0	0	0	1	0	0	0	1	1
I	0	0	0	0	0	0	0	1	0	0
J	0	0	0	0	0	0	0	0	1	0

Fig. 3. Corresponding adjacency matrix

C_d of road segment i is defined as

$$C_d(i) = \frac{\sum_j A_{i,j}}{n-1} \quad (1)$$

where $A_{i,j}$ is the entry of the adjacency matrix at row i , and column j element, and n is the network's total number of road segments. Node degree centrality measure treats all the neighbouring road segments equally and cannot distinguish instances where road segments have similar degree values.

B. Eigenvector Centrality: Contrary to node degree centrality, eigenvector centrality measures a node's importance by considering the importance of its neighbouring nodes. Therefore, eigenvector centrality considers a node as important if surrounded by important neighbouring nodes. In a social network, for example, a person connected to ten unpopular people would have lower eigenvector centrality than a person connected to the same but popular people. In

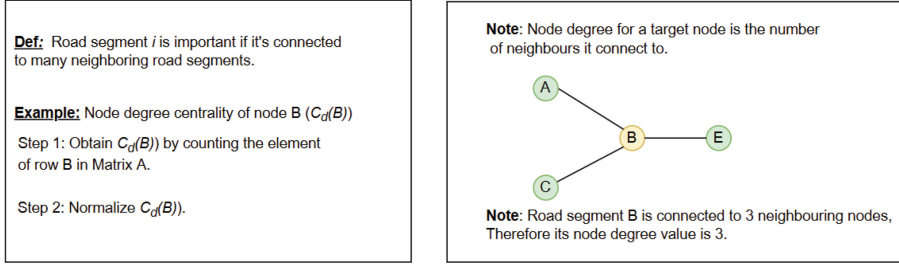


Fig. 4. Node degree centrality: (a) Node centrality processing, (b) extraction of the neighbourhood from Fig. 3.

Fig. 5, road segments B and H have the same node degree centrality (both are connected to three neighbouring road segments), however, road segment B is more popular as it's connected to neighbours with high importance measure. The eigenvector centrality C_e of road segment i is the sum of the importance of its neighbouring road segments defined as:

$$C_e(i) = \frac{1}{\lambda} \sum_{j \in N(i)} C_e(j) \quad (2)$$

where $C_e(j)$ is the importance of neighbouring road segment j , and λ is the normalisation constant. The importance of each road segment in the network is solved using the eigenvalue and the associated eigenvector equation defined as:

$$\lambda_{max} C = AC \quad (3)$$

where λ_{max} is the largest eigenvalue in the network, A is the adjacency matrix and C is the resulting eigenvector. Algorithm 1 shows the steps used in this work to compute eigenvector centrality for each road segment.

Algorithm 1. Eigenvector centrality

Input: Road network graph dataset.

Output: Road network graph dataset with C_e .

- 1: Compute the adjacency matrix A .
 - 2: Compute λ_{max} using Eq. (3).
 - 3: Compute the corresponding Eigenvector c .
 - 4: Set c as node centrality.
 - 5: **for** each road segment i **do**
 - 6: Obtain its neighbors $j \in N(i)$.
 - 7: Compute $C_e(i)$ using Eq. (2).
 - 8: **end for**
 - 9: **Return** C_e
-

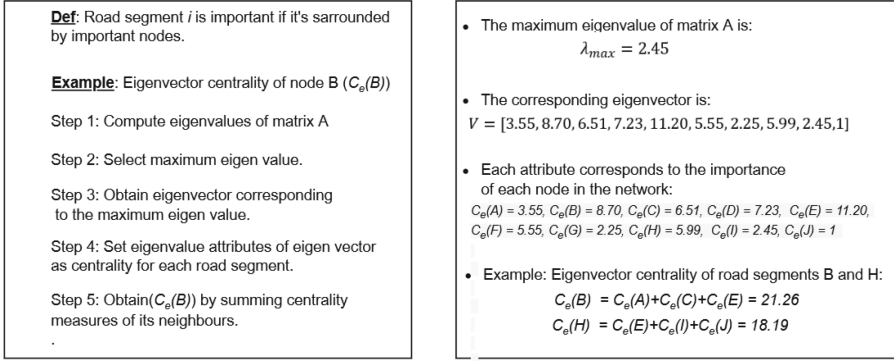


Fig. 5. Eigenvector centrality: (a) Algorithm to compute node degree centrality, (b) An example of computation of eigenvector centrality

C) Betweenness Centrality: While eigenvector centrality assesses the importance of a node based on the importance of its neighboring nodes, betweenness centrality defines node significance in a fundamentally different manner. It measures the degree to which a given node lies on the shortest paths between other nodes. In other words, it measures the role of each node in the network in allowing information to pass from one part of the network to the other. In the context of road networks, a good example of a road-type with a high betweenness centrality measure is a trunk because there is no connection between two highways without it. Depicted in Fig. 6 is an example illustrating the computation of betweenness centrality for node E. The betweenness centrality C_b of road segment i is the sum of shortest paths between any connected pair of nodes j and k that contains i , divided by the total number of shortest paths between j and k , where $j \neq k$, this is expressed as follows:

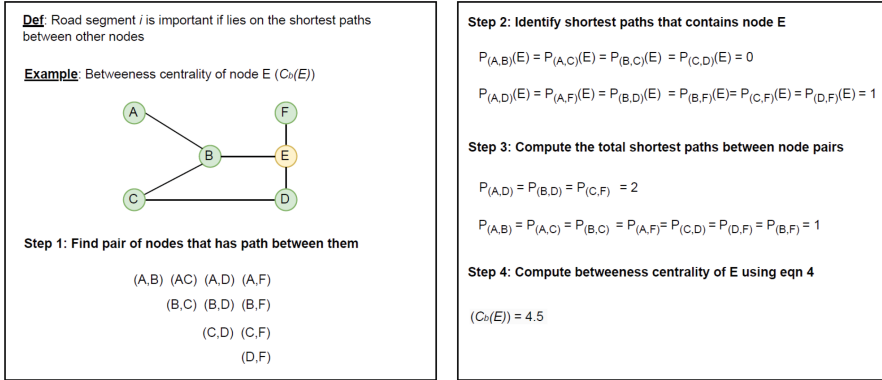
$$C_b(i) = \sum_{j \neq k} \frac{p_{j,k}(i)}{p_{j,k}} \quad (4)$$

where $p_{j,k}(i)$ is the shortest between nodes j and k that contains node i , and $p_{j,k}$ is the total shortest paths between nodes j and k . Algorithm 2 shows the steps used in this work to compute betweenness centrality for each road segment.

D) Closeness Centrality: Closeness centrality defines importance based on how close a node is to all other nodes in the network. Nodes with high closeness centrality can pass the message to all other nodes in the network relatively quickly compared to other nodes, which may require more steps. Therefore, a node is important if it has a shortest path length than all other nodes in the network. Figure 7 illustrates an example of computing the closeness centrality for node E. The closeness centrality C_c of road segment i can be computed as the inverse of the sum of the shortest paths between i and all other road segments

Algorithm 2. Betweenness centrality**Input:** Road network graph dataset.**Output:** Road network graph dataset with C_b .

- 1: Find all pairs of connected nodes.
- 2: **for** each road segment i **do**
- 3: **for** each pair of road segments j and k **do**
- 4: Compute $p_{j,k}$
- 5: Compute $p_{j,k}(i)$
- 6: **end for**
- 7: Compute $C_b(i)$ using Eq. (4).
- 8: **end for**
- 9: **Return** C_b

**Fig. 6.** Example of computation of betweenness centrality.

$j(j = 1, 2, \dots, n)$ in the network.

$$C_c(i) = \frac{1}{\sum_{j=1}^n d(i, j)} \quad (5)$$

Algorithm 3 shows the steps used in this work to compute closeness centrality for each road segment.

Structure Based Methods Structure-based methods differ from centrality methods. These methods are more related to the density metric for the whole network but focused on egocentric networks. Structure-based methods explore the structural properties of a given node by looking at the structures of its immediate neighbouring nodes.

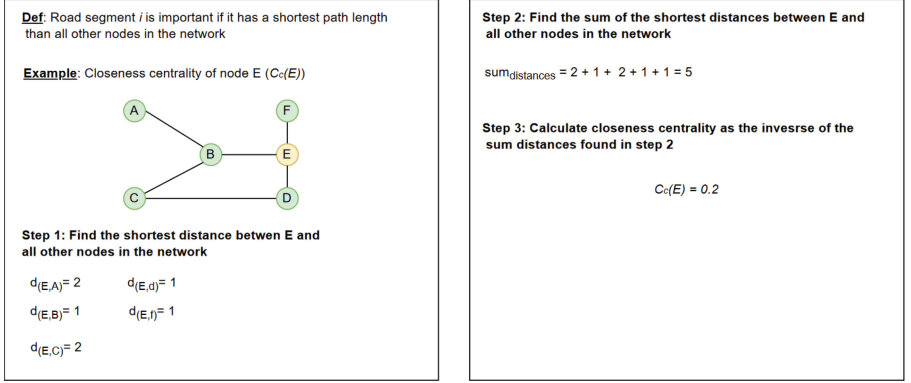
A) Clustering Coefficient: The clustering coefficient measures how connected the neighbours of a given node are to each other in the network. In a social network, for example, a person whose friends are connected to each other has a higher clustering coefficient than a person whose friends are not connected

Algorithm 3. Closeness centrality**Input:** Road network graph dataset.**Output:** Road network graph dataset with C_c .

```

1: for each road segment  $i$  do
2:   for each road segment  $j, (j \neq i)$  do
3:     Compute the shortest path between  $i$  and  $j$ .
4:   end for
5:   Compute  $C_c(i)$  using Eq. (5).
6: end for
7: Return  $C_c$ 

```

**Fig. 7.** Example of computation of closeness centrality.

to each other. The clustering coefficient C_{ce} of road segment i is calculated as the ratio of the number of edges connecting i to its neighbours to the total number of possible edges between the neighbouring nodes, this is mathematically expressed as

$$C_{cc}(i) = \frac{2n_i}{k_i(k_i - 1)} \quad (6)$$

where n_i is the degree of i and k_i is the number of edges between the neighbours of i . Algorithm 4 shows the steps used in this work to compute the clustering coefficient for each road segment.

The centrality and structure-based methods discussed above yield a 5-dimensional feature vector, and these features are used as input in training Graph Convolution Networks (GCN) model discussed in the next subsection.

3.3 Embedding with Graph Convolution Networks

Graph Convolution Networks (GCN) aims to learn an embedding function that takes high-dimensional road segment features in a non-Euclidean graph space and produces low-dimensional features in an Euclidean space. Road segment features produced by GCN can then be used to train a classifier that performs

Algorithm 4. Clustering coefficient**Input:** Road network graph dataset.**Output:** Road network graph dataset with C_{cc} .

```

1: for each road segment  $i$  do
2:   Compute  $n_i$ .
3:   Compute  $k_i$ 
4:   Compute  $C_{cc}$  using Eq. (6).
5: end for
6: Return  $C_{cc}$ 

```

road-type classification. Embedding with GCN adheres to two crucial procedures, that being (1) Locality information and (2) Aggregate and update.

Locality Information. This procedure captures the spatial relationship of a given road segment with respect to its immediate neighbours. This locality information is derived by first constructing its computation graph, which outlines the connections between a road segment and its neighbouring segments. An example of a two-hop computation graph for a target road segment, labelled 'node 1', is shown in Fig. 8. The first hop illustrates that road segments 'node 2' and 'node 3' are directly connected to the target road segment. The second hop reveals the road segments that are linked to the neighbours of the target segments.

Aggregate and Update. The aggregate and update step calculates the contribution of each neighbouring road segment to the target road segment. An aggregator function consolidates the information from the neighbours of the target road segment, and it uses it to generate its embedded vector. A two-hop computation graph, as illustrated in Fig. 8, is a three-layered representation. Road segment vectors are processed in Layer 0. In Layer 1, information is aggregated from road segments in the second layer of neighbours to those in the first layer. In Layer 2, information is aggregated from the first-hop road segments to the target road segment. At layer 0, road segment i embedding vector is derived using

$$h_v^0 = Z_i \quad (7)$$

where Z_i represents feature vector of input i . The influence of neighbouring embedded vectors of each road segment j to the current road segment i , is defined as:

$$r_i = f_{agg}(\{j | j \in N(i)\}) \quad (8)$$

where, f_{agg} represents the aggregation function, which can be (1) mean (the average of the embedded vectors of neighbouring road segments), (2) sum (the total sum of the embedded vectors of neighbouring road segments), or (3) max-pooling (the maximum value of the embedded vectors among all neighbouring

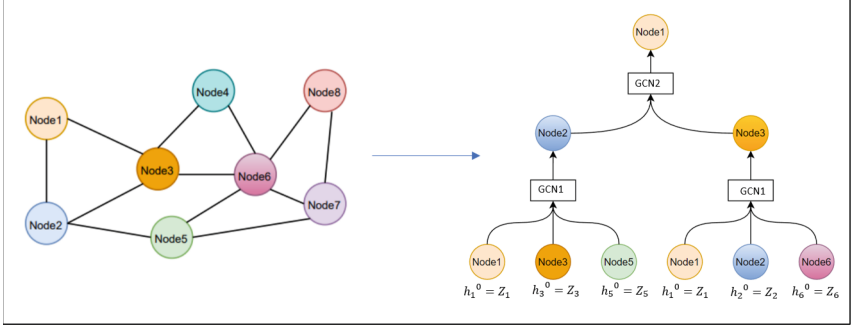


Fig. 8. Computational graph for node 1.

road segments). The impact of each aggregator function on the classification results is investigated in Sect. 4. r_i updates the embedded vector of i using

$$h_i^k = \sigma(W_k(r_i, h_i^{k-1})) \quad (9)$$

Road segment embedded vector i at the k^{th} convolution layer is denoted as h_i^k , while h_i^{k-1} represents the embedded vector of i from the previous layer. W_k refers to the learnable weight matrix, and σ denotes the activation function. Algorithm 5 [6] outlines the pseudo-code for performing the graph embedding with GCN.

Algorithm 5. Graph Embedding via Graph Convolutional Networks

Input: Road segment feature space: $RSEGS \subset \mathbb{R}^N$

Output: Road segment embedding space: $RSES \subset \mathbb{R}^M$

- 1: Define the number of hops, k
 - 2: Specify the dimensions for the input and output at each hop k
 - 3: **for** $Z_i \in RSFS$ **do**
 - 4: **Construct** the computation graph.
 - 5: **Initialize** W_k .
 - 6: **Set** h_v^0 to Z_i .
 - 7: **for** $j = 1 : k$ **do**
 - 8: **Calculate** r_i using Equation (8).
 - 9: **Update** h_v^j using Equation (9).
 - 10: **end for**
 - 11: **Extract** the embedded vector $E_i = h_v^k$
 - 12: $RSES \leftarrow RSES \cup \{E_v\}$
 - 13: **end for**
 - 14: **Return** $RSES$
-

3.4 Multilayer Perception Classifier

The features generated by the GCN embedding method are fed into a Multi-Layer Perceptron (MLP) classifier, with the Adam optimiser used for training, to perform road-type classification. The MLP classifier in this study consists of an input layer, a single hidden layer, and an output layer. The size of the input layer is determined by the dimensionality of the embedded vectors produced by the GCN embedding process. The size of the output layer matches the number of unique road-type classes. Predictions for road-type labels are generated by applying the softmax activation function to the output layer. The discrepancy between the predicted and actual road labels is measured using categorical cross-entropy loss, which is subsequently used to adjust the weight parameters of MLP via Adam optimiser.

4 Experimental Results

The road network graph dataset of Johannesburg city extracted from OpenStreetMaps (OSM) is used to conduct experiments. The experimental results are presented using the micro averaged f1-score (micro f1-score) for modelling different types of roads, using Graph Convolution Networks (GCN) embedded features obtained from structural features. Extreme class imbalances characterize the Johannesburg road network dataset. Thus, roads labels were merged and relabelled similar to the method proposed in various studies [4–6]. Table 1 shows the summary of node count per each merged and relabelled class. This section is structured as follows: Subsect. 4.1 presents the computation time of extracting road segment structural features compared to property features, and Subsect. 4.2 presents the classification results of GCN features at various aggregator functions. Subsection 3 compares the performance of the obtained results to the state-of-the-art method found in the literature, and the discussion of the experiments is presented in Subsect. 4.4.

Table 1. Datasets description [5]

Class label	Merged road-types	Node count
Class 1	highway, yes, primary, secondary, motorway-link, trunk-link, primary-link, secondary-link	3834
Class 2	tertiary, tertiary-link	1495
Class 3	road, planned, unclassified	973
Class 4	residential	8006
Class 5	living street	1853

4.1 Feature Engineering Computation Time

The computation time for generating the structural and property features is shown in Table 2. Like the state-of-the-art studies in [4–6], road segment property

features were generated using the road segment attributes such as the width, length, speed limit, geometry and number of lanes. On the other hand, structural features were generated using the methods discussed in Sect. 3.2.

Table 2. Feature engineering Computation time

Feature type	Features	Comp. time (s)
Property	width, length, geometry, speed limit, lane count	174
Structural	Node centrality measures, clustering coefficient	660

4.2 Embedding with GCN

The input to the GCN approach is a graph in which each road segment represented by a 5-dimensional feature vector. The road segments were initially split into 70% for training, 15% for validation, and 15% for testing. The GCN used in this study is a two-layer network with two hops. The first layer processes the 5-dimensional feature vectors of the road segments, while the output of the second layer is an M -dimensional embedded vector, which is then provided as input to the MLP classifier. The Adam optimiser is used with an initial learning rate of 0.0001. Several GCN aggregator functions (mean, sum, and max) and output sizes M (4, 8, and 12) are tested using the validation set to evaluate their impact on the micro F1-score. The optimal aggregator function and M parameter that achieve the highest micro F1-score on the validation set are selected and further evaluated on the test set. Regularisation and activation at each GCN layer were performed using batch normalisation and ReLU activation function, respectively. In total, 9 models are obtained from a combination of different aggregator functions and M parameters. Every model is trained for 500 epochs with batch size of 64. Figure 9 depicts the performance of each model in terms of the micro f1-score based on the validation set. Depicted in Table 3 is the parameter setting for the optimal M parameter at each aggregator function. It can be observed that the mean aggregator is the optimal aggregator function as it obtains the highest micro-f1-score (76%) based on the validation set. Subsequently, the micro f1-score achieved by the mean aggregator function is 73%. Current advancements in modelling road networks primarily utilize the Linköping city road network dataset obtained from OpenStreetMap (OSM). To ensure an equitable comparison between the proposed method and existing benchmarks, the proposed method was additionally evaluated on the Linköping City road network dataset. The results of this evaluation are presented in Table 4.

4.3 Comparison to Other Approaches

Table 5 presents a comparison between the proposed approach and existing approaches for modelling road networks based on the test set. The feature

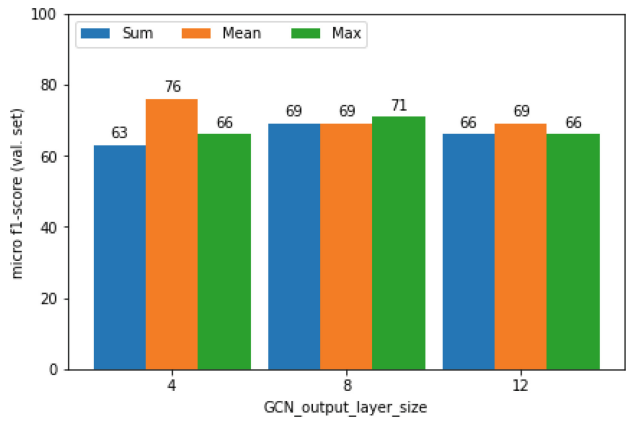


Fig. 9. Micro f1-score of various models based on validation set

Table 3. Micro f1-score at various aggregator functions: Johannesburg City

Aggregator	Optimal M	val. micro-f1 (%)	test. micro-f1(%)
Sum	8	69	73
Mean	4	76	
Max	8	71	

Table 4. Micro f1-score at various aggregator functions: Linkoping City

Aggregator	Optimal M	val. micro-f1 (%)	test. micro-f1(%)
Sum	4	67	71
Mean	8	75	
Max	8	69	

Table 5. Comparing the presented approach to existing approaches

Approach	Feature type	Features	Dataset	Test. micro-f1(%)
Raw Features [4]	Property	Width, length, speed limit, geometry	Linkoping city	59
Raw Features + DAE [5]	Property	Width, length, speed limit, geometry	Linkoping city	80
Raw Features + GCN [4]	Property	Width, length, speed limit, geometry	Linkoping city	58
Raw Features + GCN + DAE [6]	Property	Width, length, speed limit, geometry	Linkoping city	75
Raw Features + GAIN [4]	Property	Width, length, speed limit, geometry	Linkoping city	81
Structural Features + GCN [proposed]	Structural	Degree centrality, eigenvector centrality, betweenness centrality, closeness centrality, clustering coefficient	Linkoping city	71

engineering stage significantly differs between the presented approach and other approaches. Existing approaches obtain each road segment feature vector using road property features including segment width, length, speed limit and geometry. Meanwhile, the approach proposed in this study uses road segment structural features such as centrality measures to generate a feature vector for each road segment. Raw features refer to using property features and MLP classifier for modelling a road-type classification. Raw features + DAE refers to embedding the raw features with the Deep AutoEncoder (DAE) model before achieving the classification task with the MLP classifier. Raw features + GCN refers to embedding the raw features with the Graph Convolution Networks (GCN) model before achieving the classification task with the MLP classifier. Raw features + GCN + DAE refers to embedding the raw features with GCN and DAE models. Raw Features + GAIN refers to embedding the raw features with Graph Attention Isomorphism Networks (GAIN). Finally, Structural Features + GCN [proposed method] refers to embedding the structural features with GCN. The results in Table 5 indicate that modelling road-type classification with structural features + GCN significantly outperforms the results obtained by raw features only (59%) and raw features + GCN (58%). The results obtained by the proposed method fall short by only 4% compared to the Raw Features + GCN + DAE method. Furthermore, the results obtained by the proposed method fall short by 9% and 10% compared to Raw Features + DAE and Raw Features + GAIN methods, respectively.

4.4 Discussion

The computation cost of extracting road segment features using the node structural methods proposed in this work is compared to that of extracting property features using road segment attributes in Table 2. It can be observed that extracting features with the proposed structural methods takes more time than extracting property features. This observation is reasonable since structural-based methods often require considering the neighbours of the target road segment and the entire network to generate features. On the other hand, property features are generated independently of neighbouring road segments (e.g. obtaining road segment width does not rely on the width of neighbouring road segments). The hyperparameter settings of aggregator functions and the GCN output size M are depicted in Table 1. It can be observed that the mean aggregator function and the M parameter of 4 dimensions are the optimal parameters with the validation and test micro f1-score of 76% and 73%, respectively. The results indicate that the GCN generates robust embeddings when each road segment is presented using the mean vector of its immediate neighbours. Additionally, the proposed method was evaluated on the Linköping city road network dataset, and the results are presented in Table 4. It can also be observed that the mean aggregator function and the M parameter of 8 dimensions are the optimal parameters with the validation and test micro f1-score of 75% and 71%, respectively. The results obtained by the proposed method on Linköping city

road network data were compared with state-of-the-art methods in Table 5, and the following observations can be made.

- The proposed method achieves better performance than the approach that relies solely on raw features.
- The presented method surpasses the results achieved by the raw features combined with the GCN method.
- The proposed method achieves comparable results with raw features + DAE+ GCN method.
- The proposed method falls behind raw features + DAE and raw features + GAIN methods

The discrepancies observed between the proposed method and existing approaches can be attributed to several methodological variations; these include feature Dissimilarity: The proposed method adopted structural features for road segment representation during training. Conversely, existing methods within the literature primarily utilized road segment property features. This difference in feature selection can lead to distinct model behaviours and results.

Feature dimensionality: The dimensionality of the structural features produced by the proposed method is significantly smaller than that of the road-segment property features. Small features may not provide enough information to discriminate between different classes or categories in the data effectively. This can result in a lack of predictive power and accuracy in the model's predictions. Furthermore, features in smaller dimensions may not capture the complexity of the underlying data distribution, making it challenging for the model to learn intricate patterns and relationships within the data. This limitation can hinder the model's ability to make accurate predictions, especially in tasks that require a deep understanding of the data. Conversely, features in a higher-dimensional space raise the risk of overfitting, where the model may learn insignificant patterns in the training data that fail to generalize well to new data. Overfitting can lead to poor performance on new data and reduce the model's predictive accuracy.

Different Graph Embedding Methods: The proposed method uses GCN, while other approaches in the literature use DAE and GAIN. It is crucial to emphasize that the proposed study does not advocate for the complete abandonment of property features in road network modelling. Instead, it highlights the potential benefits of complementing such features with structural features, particularly where road segment attribute data is unavailable.

5 Conclusion

Existing road-type classification methods rely mainly on road segment attributes including width, length, speed limit, and geometry to generate road segment property features, and very accurate classification results can be obtained. However, road attributes obtained from the publicly available mapping platform,

OpenStreetMaps (OSM), are often missing, thus making a road-type classification with road attribute features very unreliable. This work proposes using structural-based feature methods for road segment feature generation, and the experimental findings demonstrate that the presented approach can be used to model road network datasets where road attributes are missing. Future work will attempt to combine structural and road attribute features to enhance the classification results for road-type classification tasks.

References

1. Deekshetha, H.R., Madhav, A., Tyagi, A.: Traffic Prediction Using Machine Learning, pp. 969–983 (2022)
2. Masiero, L., Casanova, M., Tilio, M.: Travel time prediction using machine learning. In: Proceedings of the 4th ACM SIGSPATIAL International Workshop on Computational Transportation Science, pp. 34–38 (2011)
3. Song, X., Zhang, C., Yu, J.J.: Learn travel time distribution with graph deep learning and generative adversarial network. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pp. 1385–1390 (2021)
4. Gharaee, Z., Kowshik, S., Stromann, O., Felsberg, M.: Graph representation learning for road-type classification. *Pattern Recogn.* **120** (2021)
5. Molefe, M.E., Tapamo, J.R.: Road-type classification with deep autoencoder. *Comput. Intell. Neuroscience* (2023)
6. Molefe, M.E., Tapamo, J.R.: A new approach for road-type classification using multi-stage graph embedding method. In: Rodríguez-González, A.Y., Pérez-Espinosa, H., Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A., Olvera-López, J.A. (eds) *Pattern Recognition. MCPR 2023. Lecture Notes in Computer Science*, vol. 13902 (2023)
7. Musfira, I., Padraig, C., Michela, B.: Automated highway tag assessment of OpenStreetMap road networks. In: the 22nd ACM SIGSPATIAL International Conference, pp. 449–452 (2014)
8. Liu, T., Lunga, D.: Automated openstreetmap data alignment for road network mapping. In: IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium, pp. 2603–2606 (2020)
9. Vargas-Munoz, J.E., Srivastava, S., Tuia, D., Falcão, A.X.: OpenStreetMap: challenges and opportunities in machine learning and remote sensing. *IEEE Geoscience Remote Sens. Mag.* **9**(1), 184–199 (2021)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks (2016)
11. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs (2017)
12. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks (2017)



Leveraging Machine Learning and Climate Data for Enhanced Annual Crop Production Forecasts in Senegal

Kpangni Alex Jérémie Koua^{1,3}(✉) , Fatou Cire Emilie Sané¹ ,
Michael Warren², and Jonathan Mwaura²

¹ University Gaston Berger of Saint-Louis, UFR Applied Science and Technology,
Saint-Louis, Senegal

{koua.kpangni-alex-jeremie,sane.fatou-cire-emilie}@ugb.edu.sn

² Roux Institute At Northeastern University, Portland, ME 04101, USA

{warren.mic,j.mwaura}@northeastern.edu

³ University Jean Lorougnon Guédé of Daloa, UFR Environment,
Daloa, Côte d'Ivoire

Abstract. Accurate crop production forecasting is crucial for enhancing food security and guiding agricultural policies in Senegal, a country where agriculture plays a vital role in the economy. Traditional methods, employed by the International Production Assessment Division (IPAD) of the U.S. Department of Agriculture, have shown limitations, particularly in predicting the productions of key crops like rice, corn, millet peanuts, and sorghum. To address these challenges, this study presents a machine learning-based approach integrating climate data rainfall, temperature, and Normalized Difference Vegetation Index (NDVI) to improve forecasts. The methodology applies advanced statistical techniques, including differencing, to remove trends from the data, and various regression models. The work also introduces a feature weighting strategy that accounts for regional differences in crop production volumes, which has proven effective in reducing the complexity of the model while preserving essential information. The results show a significant accuracy improvement, with models achieving a Mean Absolute Percentage Error (MAPE) of 6.4% in 2022 and 7.6% in 2023, cutting errors by 60% compared to IPAD. These findings underscore the potential of machine learning techniques to significantly enhance crop production forecasting in Senegal, providing a robust tool for policymakers to make data-driven decisions regarding agricultural practices, import/export strategies, and pricing policies.

Keywords: Machine learning · Production forecasts · Climate data · Senegal

Supported by the United States Department of State (DoS): Embassy in Dakar, under the University Partnership Initiative, Grant No. SSG20023GR0032.

1 Introduction

Senegal, like many countries, is highly dependent on agriculture [1]. Agriculture plays a central role in Senegal's economy, providing employment and serving as the primary source of food security [2]. Consequently, more than one-eighth of the country's national territory is dedicated to agriculture, making it an essential driver of rural livelihoods and economic development [3]. Given this heavy reliance, accurate and timely forecasts of annual agricultural production are critical. Such forecasts enable the government to guide agricultural policies, manage food reserves, and ensure food security for the population. Moreover, these forecasts influence decisions related to imports, exports, and domestic food distribution, directly affecting the livelihoods of farmers, traders, and consumers. Precise forecasts help stabilize food prices, reduce the risk of food shortages, and enhance the planning of agricultural investments [4].

Despite the critical importance of agricultural production forecasts, achieving accuracy remains a significant challenge in Senegal. Traditional forecasting methods, while useful, often face limitations due to the complexity of factors that influence agricultural outcomes. One of the primary challenges is climate variability, which has become increasingly unpredictable in recent years. Rainfall patterns, in particular, have a profound impact on agricultural production, as the majority of farming in Senegal is rain-fed [5]. Small-scale farmers, who constitute the bulk of the agricultural workforce, are especially vulnerable to these fluctuations [6]. They often lack access to modern agricultural inputs such as irrigation systems, mechanization, fertilizers, and pesticides. As a result, they have limited ability to control the growing conditions of their crops, making them highly dependent on natural environmental conditions [7]. This study addresses the crucial problem of the inaccuracy of agricultural production forecasts in Senegal.

The International Production Assessment Division (IPAD) of the U.S. Department of Agriculture regularly publishes agricultural production forecasts for Senegal, as well as for other countries in the region. IPAD utilizes satellite imagery, weather data, and other remote sensing techniques to predict crop productions. These forecasts have proven useful in providing a broad overview of agricultural trends. However, in recent years, there has been growing concern that IPAD's forecasts for Senegal, particularly for staple crops such as rice, peanuts, among others, have been underperforming¹. One hypothesis is that Senegal's unique agricultural system is not receiving sufficient attention in IPAD's annual forecasts. Factors such as crop diversity, local farming practices, environmental impacts, and the latest advances in data science are not always fully integrated into these forecasts, leading to potential inaccuracies [8].

In light of these challenges, there is a pressing need to develop more accurate forecasting models tailored to Senegal's specific agricultural context. The work described in this paper aims to fill the gap by applying machine learning (ML) techniques to enhance crop production forecasts. It integrates comprehen-

¹ <https://ipad.fas.usda.gov/countrysummary/default.aspx?id=SG>.

sive climate data to improve predictions for key staples like rice, corn, millet, peanuts, and sorghum. ML has the potential to significantly enhance forecast accuracy by capturing complex interactions between variables and making predictions based on large datasets. These models can analyze vast amounts of climate data, including temperature, precipitation, and Normalized Difference Vegetation Index (NDVI), a measure of vegetation health [9,10]. By doing so, they can better account for the complex interplay of environmental factors that influence agricultural production.

The goal of this study was to conduct a comparative analysis between the forecasts generated by the investigated ML models and those produced by IPAD. By doing so, the study aimed to identify areas where traditional forecasting methods may be falling short and propose recommendations for improving the accuracy of future forecasts. Specifically, the discussed work evaluates the performance of different model groups, including regression models, time series models, and ensemble models, using key climate indicators such as rainfall, temperature, and NDVI. The models are trained on historical data from Senegal to predict the productions of the selected crops.

Improving forecast accuracy could significantly impact agricultural policy in Senegal. Reliable forecasts would enable policymakers to better anticipate food production shortages or surpluses, optimizing import, export, and distribution decisions. For farmers, an improved forecasting system would support planning of crop cycles, enhancing resilience to climate variability. This research aims to lay the groundwork for innovative forecasting tools that contribute to the long-term sustainability and food security of Senegal's agricultural sector [11].

The remainder of this paper is organized as follows: Sect. 2 provides a review of related work, focusing on agricultural forecasting and the role of ML. Section 3 outlines the methodology followed in work, while Sect. 4 presents and discusses the experimental results. Finally, Sect. 5 concludes the paper with a summary of the findings and insights.

2 Related Work

Agricultural production forecasting has long been a focus of extensive research, given its critical importance for ensuring food security, economic stability, and informed policy decisions [12]. Over the years, a variety of methods and models have been proposed, each evolving with advancements in data collection and computational technologies. Traditionally, statistical approaches such as linear regression and time series analysis have been widely used for crop production forecast [13]. These models rely on historical data and basic relationships between variables like rainfall, temperature, and crop output to generate forecasts. While they have proven effective in many scenarios, these methods often fall short when it comes to capturing the complexities and nuances of modern agricultural systems, particularly as the scale of data grows and climate variability increases [14]. This limitation has driven researchers to explore alternative approaches, especially those enabled by machine learning (ML) and artificial intelligence (AI) technologies [13,15].

In many cases, agricultural data is time series data, a sequence of data points collected at regular time intervals, such as daily rainfall, monthly crop yields, or annual temperature fluctuations. Time series data has a natural temporal ordering, which is crucial for capturing trends, seasonality, and cyclical patterns.

In agriculture, time series analysis is essential for modeling how factors such as climate, production and soil conditions evolve over time, helping to predict future trends as carried out in [13, 16]. For example, crop yields may depend on the cumulative impact of rainfall, temperature, and soil health over several seasons. The temporal structure of time series data makes it ideal for forecasting, but it also requires more sophisticated techniques to handle issues like seasonality, autocorrelation, and long-term trends, which simple models may overlook.

ML algorithms excel at identifying complex patterns in data that are often missed by traditional methods. Recent studies have demonstrated that these algorithms hold significant promise for improving the accuracy of crop yield forecasts.

Machine learning models such as Random Forest (RF) [17] are particularly well-suited for this task because they can handle large datasets, reduce overfitting, and better capture the intricate relationships between variables. In [17], the authors used various ML techniques as RF, Support Vector Machines (SVM), Gradient Descent (GD), Long Short-Term Memory (LSTM), and LASSO regression (LR) to estimate crop yield in the Rajasthan state of India, focusing on five identified crops. Among the algorithms tested, the RF technique produced the best results, with an R^2 of 0.963, an RMSE of 0.035, and an MAE of 0.0251. These performances were validated using cross-validation techniques. This work applies the crop selection method to help farmers address crop yield challenges.

In [3], the study examines the predictability of yields for key staple crops in Senegal (peanuts, corn, millet, and sorghum) using four advanced regression methods. These methods include three commonly used machine learning algorithms: SVM, RF, and Neural Network (NN), as well as a multiple linear regression method, the Least Absolute Shrinkage and Selection Operator (LASSO). The models used climate and vegetation data as inputs. The optimized models were applied to the testing data, and the predicted R^2 values were calculated to assess their performance. The results showed that the predictions obtained using the linear model (LASSO) were generally lower than those from the other methods (SVM, NN, and RF), except in the case of millet, where LASSO provided better predictions than RF.

Jorvekar et al. [18] present a comparative study evaluating the performance of various regression models for agricultural crop yield prediction. A comprehensive dataset was collected, encompassing historical crop yield data, weather parameters, and pesticide data from different agricultural regions. The study applied and compared multiple regression models, including Linear Regression (LR), K-Nearest Neighbor Regression (KNN), Support Vector Regression (SVR), Decision Tree Regression (DTR), Random Forest Regression (RFR), Gradient Boosting Regression (GBR), as well as LASSO Regression, ElasticNet Regression, and Ridge Regression, to predict yields for different crops. The results offer

insights into the comparative performance of these models, highlighting that the effectiveness of each model varies depending on the crop type, region, and dataset used.

Work reported in [19], utilized time series and ensemble techniques to forecast banana crop yield in Tanzania. The models employed included Seasonal AutoRegressive Integrated Moving Average (ARIMA) with Exogenous Variables (SARIMAX), State Space (SS), and LSTM. An ensemble model was formulated using a weighted average approach, combining these techniques to enhance prediction accuracy. Their results highlighted the effectiveness of these models, with statistical metrics validating the reliability of the forecasts. This approach could potentially be applied to predict other crop yields under similar conditions. Statistical models such as ARIMA are still widely used in the field [16]. These ensemble methods combine the predictive power of multiple models to create a more robust and accurate forecasting tool, making them increasingly popular in agricultural forecasting [20–23].

In addition to ensemble methods, deep learning techniques have emerged as powerful tools for modeling time series data, which is critical in agricultural production forecasting. Recurrent Neural Networks (RNN) and LSTM networks are particularly effective for capturing temporal dependencies in crop yield data. These models are designed to remember long-term dependencies, which is crucial when dealing with seasonal variations and long-term climate trends that impact agricultural output. Their ability to model non-linear relationships and adapt to changing conditions makes them valuable for forecasting in dynamic environments like agriculture [24, 25].

IPAD plays a critical role in providing agricultural forecasts in Senegal and other countries. IPAD's models incorporate a mix of statistical and satellite data to provide timely forecasts of agricultural production. However, on the IPAD website, the forecasts provided show that there is still room for improvement when it comes to Senegal. The complexity of local farming practices, crop diversity, and environmental factors can pose significant challenges for general forecasting models. As a result, this study has increasingly focused on comparing IPAD's forecasts with alternative methods, particularly machine learning models. These comparisons reveal that machine learning approaches outperform traditional statistical methods in certain contexts, especially when dealing with non-linear patterns and large datasets [14, 26].

3 Materials and Methods

This section outlines the approach utilised in the implementation and modelling of the crop production forecast system reported in this work (see Fig. 1). The study utilised a comprehensive model training process designed to exploit various sources of agricultural data for predictive modeling.

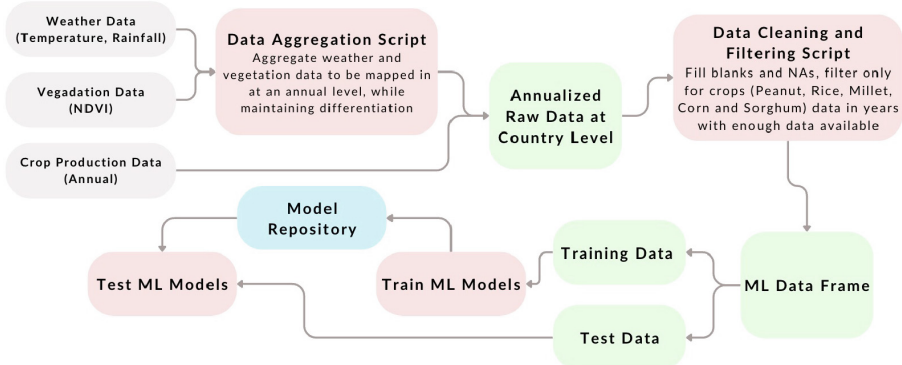


Fig. 1. Model Training Process

3.1 Description of Data

Source of Data: For this study, Crop production data, temperature data, rainfall data and NDVI data are used from 1961 to 2024 with various source.

- **Crop Production Data:** Crop Production data is real data from FAOSTAT of the Food and Agriculture Organization of the United Nations (FAO). FAO-STAT provides free access to food and agriculture statistics (including crop, livestock, and forestry subsectors) for more than 245 countries and territories and covers all FAO regional groups from 1961 to the most recent year available. This work used production data for the various crops grown in Senegal such as millet, peanuts, corn, rice, and sorghum. They were selected because of their high use and availability in several departments of Senegal [3]. Understanding crop production dynamics helps in determining the relationship between environmental factors and production variability.
- **Temperature Data:** The temperature data used in this study were obtained from METEOSTAT. Meteostat provides detailed weather data for thousands of weather stations and locations worldwide. The Meteostat Python library gives a simple API for accessing open data. In the reported work, dataset includes data from 11 stations in Senegal between the years 1973 to 2023. Temperature affects crop development phases, including germination, flowering, and grain filling. Rising temperatures may result in production declines across agronomic species, potentially reducing crop productivity. Monitoring temperature trends helps identify stress periods for crops, influencing decisions on planting schedules and crop types.
- **Rainfall Data:** These data are available from between the years 1996 to 2024 and was collected from FEWSN and RFE, the Famine Early Warning Systems Network (FEWS NET). FEWS NET is a leading provider of early warning and analysis on acute food insecurity around the world. RFE is Rainfall Estimate. On this source the rainfall data are obtained by province for Senegal. Rainfall is critical for agricultural productivity, especially in rainfed farming

systems. Variability in rainfall patterns directly influences crop growth and production, making it a key factor in predicting agricultural outcomes. In regions like Senegal, where rainfed agriculture is predominant, fluctuations in water availability due to changing rainfall can significantly impact crop productions.

- **NDVI (Normalized Difference Vegetation Index) Data:** This study used the vegetation data from FAO-ASIS between the years 1984 to 2024. FAO is a Food and Agriculture Organization of the United Nations and ASIS is the Agricultural Stress Index System. ASIS was developed for the early identification of agricultural areas probably affected by dry spells, or drought in extreme cases by FAO. ASIS gives satellite data of NDVI and holds annual and multi-annual data archive (back to 1984). NDVI measures vegetation health and biomass by capturing plant growth using spectral bands. High NDVI values indicate healthy, dense vegetation, while low values suggest poor plant health. NDVI data is useful for monitoring crop growth stages, assessing the impact of drought, and predicting production. The study used monthly maximum NDVI images to classify crop types and evaluate the health of crops, providing valuable insights into vegetation patterns and their relationship with production.

The complete dataset contains fourteen regions namely Dakar, Diourbel, Fatick, Kaolack, Kolda, Louga, Matam, St Louis, Tambacounda, Thiès, Ziguinchor, Kaffrine, Kédougou, Sédhiou with crop production, vegetation, rainfall and temperature data. Further, five major food crops (rice, corn, millet, peanuts, and sorghum) consumed in Senegal between the years 1995 to 2023 were selected for crop production analysis. The integration of rainfall, temperature, NDVI, and crop production data is crucial for production forecasts. Rainfall and temperature affect crop growth, while NDVI provides real-time insights into vegetation health. Together, these datasets allow for accurate predictive modeling of crop production, enabling better planning and decision-making for agricultural management under changing climatic conditions.

Data Pre-processing: The process applying for each dataset (vegetation, rainfall, temperature) is designed to process data by performing various preprocessing steps, including loading, cleaning, filtering and pivoting for simplify analysis and comparaisn accross times and space. Additionally, the actual production data for 2023 is manually entered for the crops (peanuts, corn, millet, rice, and sorghum). The manually pulled 2023 data serves as a validation or comparison point for model predictions. Data about weighted averages was also added, based on crop production from different provinces. This function calculates the weighted average of a specific metric (Vegetation, Rainfall, Temperature) for each crop, using the production percentage of each crop from different provinces. The purpose of this weighting is to account for how much each province contributes to the total production of a given crop when calculating the metric values. The weighted average calculation is critical to ensure that metrics, such as the NDVI, reflect the relative importance of each province for each crop. A province that

contributes more to the total production of a crop will have a higher influence on the weighted average. Figure 2 is an example which illustrates the weighted average calculation process for all Senegalese provinces, using temperature as the metric.

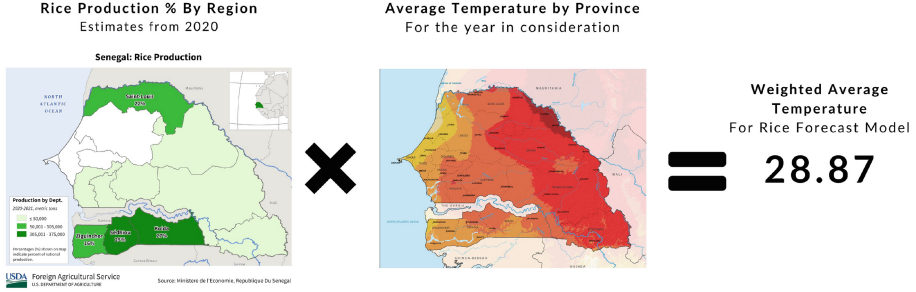


Fig. 2. Illustration of Weighted Average Calculation

Agricultural data, especially crop production data, often exhibit trends due to factors like technological advancements, changes in farming practices, or climate changes over time. These trends can distort the relationship between variables, making it harder for models to accurately predict future production. This work used Pearson correlation to help identify and quantify the linear relationship between crop production and Climate changes (Temperature, Rainfall and NDVI). The data captured the observations of climate changes in chronological order, where each data point is associated with a particular timestamp. This temporal aspect is specific for time series data. As such, differencing is used to eliminate the long-term trend, allowing the model to focus on other cyclical patterns or short-term variations that might affect crop production.

Pearson Correlation: Crop Production vs. Climate Features:

Pearson correlation is a statistical technique used to measure the degree of relationship between two variables (e.g., understanding how temperature affects crop production). It is the most popular and widely used Pearson correlation coefficient (denoted as r). Pearson correlation coefficient takes values between -1 and 1, where 1, 0, and -1 indicate a perfect match, no correlation, and perfect negative correlation, respectively [28]. Mathematically, given paired data $\{(x_1, y_1), \dots, (x_n, y_n)\}$ consisting of n pairs, the Pearson correlation is calculated as follows:

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}, \quad (1)$$

where x_i and y_i are the variables being compared, $\bar{x}$ and $\bar{y}$ represent their respective means with $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ and analogously for $\bar{y}$.

Also, Eq. (1) is applied to compute the Pearson correlation coefficient between crop production and various climate features (such as temperature,

rainfall, vegetation). The correlation data in Table 1 highlights several relationships, expressed as percentages.

Table 1. Pearson Correlation

Crop	Rainfall	Temperature	NDVI
Rice	63%	8%	38%
Corn	54%	-7%	38 %
Millet	66%	33%	59%
Sorghum	51%	-5%	49%
Peanuts	52%	36%	38%

One of the key insights is the strong correlation between rice production and rainfall, which is logical given the heavy dependence of rainfed rice plantations on water availability. Peanuts and Millet production appear to be more impacted by temperature than other crops. Overall, rainfall appears to be the most important factor in crop production. The Pearson correlation plays a central role in this study, quantifying the linear relationship between agricultural production (rice, maize, millet, sorghum, peanuts) and climatic variables (rainfall, temperature, NDVI). Calculating these correlations is crucial for identifying the most relevant features, as a high correlation indicates a stronger contribution to the model’s performance. This analysis helps in selecting the best features to build an accurate and efficient model.

To analyze the relationships between the variables (rainfall, temperature, NDVI and crops) in detail, Pearson’s coefficient table was used to create a heat map of the data, making it easy to visualize the strengths and directions of the correlations. Figure 3 shows the correlation coefficients between crops (rice, maize, millet, sorghum, peanuts) and climatic variables (rainfall, temperature, NDVI). The colors make it easy to visualize strong positive correlations (warm colors, red/orange) such as rain-millet and weak or negative correlations (cold colors, blue) such as temperature-maize. This representation facilitated the interpretation of key relationships to improve forecasts.

Data Differencing: Differencing is a key technique used to remove trends and make time series data stationary before applying machine learning models. To apply differencing, the current value is subtracted from the previous value in the data series. This helps in de-trending the data, which is crucial when making predictions using models like ARIMA, or even when feeding the data into regression or Random Forest models.

By detrending the crop production data, models such as ARIMA and others can capture the more relevant dynamics of production, improving predictive accuracy. Figure 4 illustrates the differencing applied to rice production between 1997 to 2023. On the other hand, differencing was used to detect outliers through

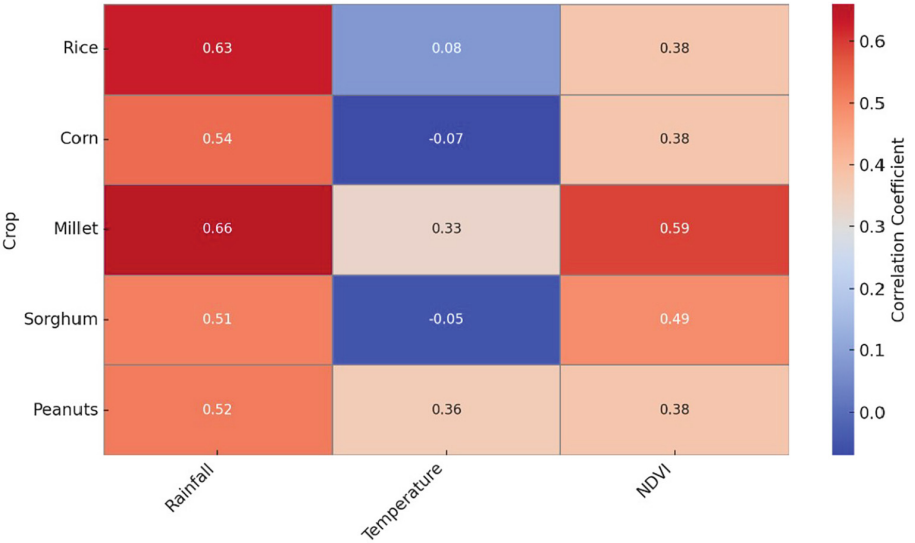


Fig. 3. Pearson Correlation Between Crops and Variables

an analysis of variations between successive data, and weighted averages to minimize their impact by applying low weights to values detected as outliers, while retaining their adjusted contribution in the model, thus stabilizing the results.

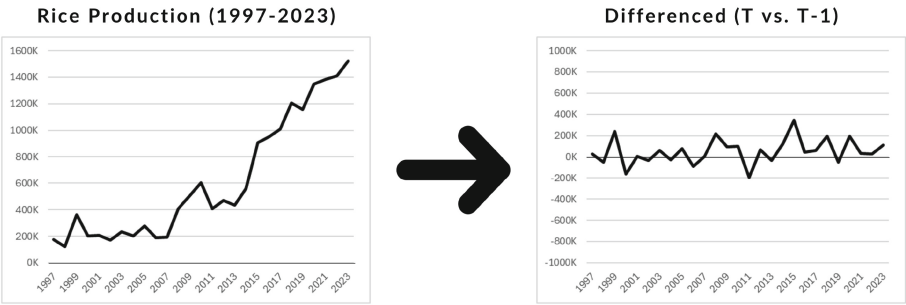


Fig. 4. Differencing of Rice Production Data

3.2 Modeling, Training and Testing

Models. This study used a Baseline model as a reference point. In this context, the basis is a simple differentiation (i.e., calculating the variation in crop production from one year to the next), without modeling the relationships between crop production and characteristics(temperature, Rainfall, NDVI). This model

is useful for understanding whether more advanced models provide added value compared with a simple differentiation approach. It can give insight into the stability of trends or changes in production [29]. Various widely-used machine learning methods were applied to predict crop production (rice, corn, millet, sorghum, and peanuts) in this study. These included Linear Regression, along with other models that incorporate regularization, such as LASSO Regression and Ridge Regression. These regression techniques are particularly effective for handling multicollinearity and selecting important features in predictive modeling. The use of these linear models is explained by the strong linear relationship between the variables, as demonstrated by the Pearson correlation. In addition to these, Random Forest, a robust ensemble method known for its ability to manage outliers and feature noise, and ARIMA, commonly used with time series data for forecasting. These methods have been extensively applied in similar agricultural contexts [3,27].

The parameters considered in this study include training data years, differencing, moving average (MA), and LASSO regularization. Table 2 shows the different test values for each parameter. Each of these plays a crucial role in shaping the models used to predict crop production.

Table 2. Summary of model parameters and training settings

Parameter	Value
training_data_years	10, 15, 20
differencing	0, 1
ma	0, 2
LASSO	0.1

In terms of performance, regularized models like Lasso and Ridge are well-suited to data with numerous features, as they reduce overfitting by penalizing coefficients. In contrast, models like Random Forest and ARIMA are often better suited to time series or nonlinear data, as they capture complex dependencies and temporal patterns more effectively, thus improving performance in these specific contexts.

Training Data Years: This parameter refers to the number of historical years used to train the models. In time series forecasting and regression analysis, selecting the appropriate length of training data is critical to capturing long-term trends while avoiding overfitting to specific periods. The models are trained using **20** years of historical data.

Differencing: It is used to make the data stationary by removing trends or seasonality. This ensures that the underlying patterns are more stable over time, enhancing the model’s ability to make accurate predictions. In this study, a differencing order of **1** was used to transform the data into a stationary format.

Moving Average (MA): The MA parameter in the ARIMA model controls the number of lagged forecast errors used to model the time series. In this study, the MA value was set to 0, indicating that the model does not incorporate previous forecast errors but focuses on the autoregressive and differencing components for prediction.

LASSO Regularization (α) : In LASSO regression, model performance was evaluated for different values of the hyperparameter (alpha), and the best-performing value considered optimal was $\alpha=0.1$. This introduces a moderate level of regularization, shrinking some coefficients while retaining the most important features for predictive accuracy.

These parameters were selected to optimize the model's performance and ensure accurate predictions of crop production. This study models agricultural production using Pearson correlation to analyze relationships between production and climatic factors, and employing various Machine Learning models to predict future production. It combines several techniques to manage over- and under-learning, focusing on appropriate model selection, regularization, temporal cross-validation and data pre-processing.

Training and Test: The approach employed in this work used advanced temporal cross-validation methods [27]. This technique, shown by Fig. 5, is specially adapted for time series data. A sliding window of 20 years for training is used, followed by a single year for testing, repeated over seven consecutive iterations. This method, often referred to as walk-forward validation, allows to evaluate the model's robustness and generalization over time. The analysis begins with a training period from 1997 to 2016, followed by testing on the year 2017. Subsequently, the training window is shifted by one year, with the next iteration training on data from 1998 to 2017 and testing on 2018. This process is repeated, advancing the training period by one year in each step. The final iteration involves training from 2003 to 2022, with testing conducted on 2023. This approach offers several key advantages. First, it closely simulates real-world prediction scenarios, where historical data is used to forecast the near future, reflecting the practical challenges of predictive modeling. Additionally, it allows for a comprehensive evaluation of the model's stability as it encounters temporal changes in agricultural data, ensuring that the model adapts to shifts in patterns over time. By incorporating annual variations into the testing process, this method provides a more robust assessment of model performance. Furthermore, it enables the detection of potential trends or changes in prediction accuracy, helping to identify periods where the model may perform better or worse, and offering insights for future improvements. Furthermore, this method allows to analyze how climate changes or evolving agricultural practices over this 27-year period (1997–2023) may affect the accuracy of the predictions.

4 Results and Discussion

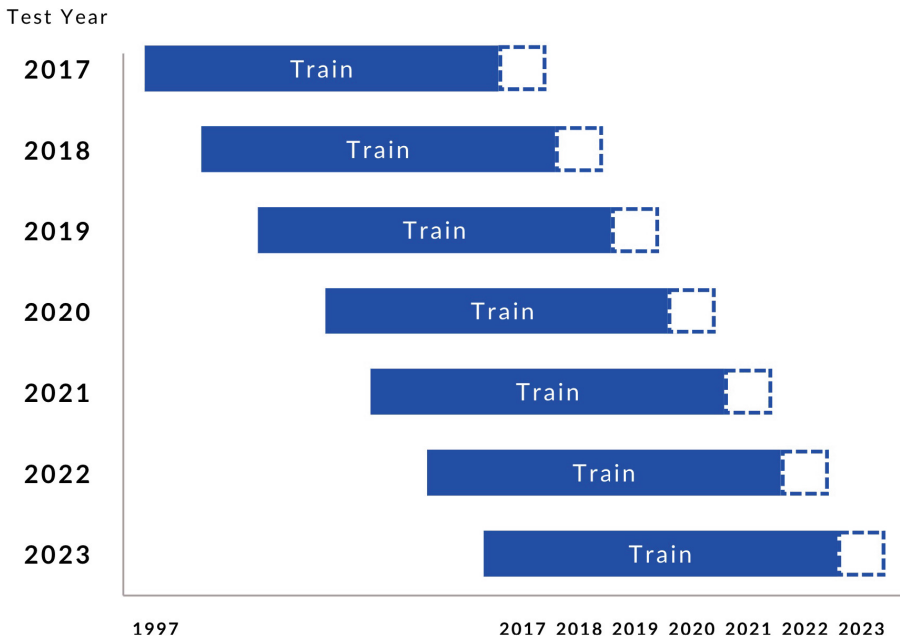


Fig. 5. Training and Test periods

This section provides the results obtained using the previously mentioned ML techniques. The obtained results are compared to those reported by IPAD (i.e., forecasts). This study uses the Mean Absolute Percentage Error (MAPE) metric to evaluate the accuracy of various forecasting models. It measures the average percentage error between predicted and actual values, providing a clear understanding of the relative discrepancy between predictions and observations. The formula for **MAPE** is as follows:

$$\text{MAPE} = \frac{1}{n} \sum_{t=1}^n \left| \frac{Y_t - \hat{Y}_t}{Y_t} \right| \times 100, \quad (2)$$

where n is the total number of data points (observations), Y_t is the actual value at time t , $\hat{Y}_t$ is the predicted value at time t , $|\cdot|$ denotes the absolute value.

Table 3 presents the results of this study's approach, which relies on time series analysis for agricultural production forecasting.

Table 3 displays the MAPE for different model types, ranging from simple linear regression to more complex time-series models such as ARIMA. The obtained

Table 3. MAPE Performance

Model Type	MAPE
Linear Regression	11.84 %
LASSO Regression	12.17%
Ridge Regression	12.42%
Baseline - Differencing Only	12.70%
Random Forest	13.22%
ARIMA (1 AR, 1 Diff)	13.86%
ARIMA (3 AR, 1 Diff)	16.05%
Linear Regression (No Differencing)	44.59%

results demonstrate that Linear regression achieved the lowest MAPE at 11.84%, making it the best-performing model for this dataset. This simpler regression model proves most effective, striking a balance between complexity and performance for this specific agricultural forecasting task.

LASSO regression (12.17%) and Ridge regression (12.42%) show slightly higher MAPE values compared to linear regression, suggesting that regularization does not significantly enhance performance in this case. The Baseline - Differencing Only model (12.70%) serves as a reasonable benchmark, though it performs slightly worse than the regression models.

More complex models, such as Random Forest (13.22%) and ARIMA (13.86% and 16.05%), perform worse than simpler linear models. This may indicate that the relationship between features and the target variable is relatively linear. The ARIMA model results suggest that adding more autoregressive terms (3 AR versus 1 AR) does not improve forecasts, possibly reflecting limited long-term dependencies in the data.

The stark difference between linear regression (11.84%) and undifferentiated linear regression (44.59%) underscores the critical importance of proper preprocessing for time-series data, likely due to trends or seasonality in the agricultural data.

Figures 6, 7, 8, and 9 provide a comparative analysis of crop production forecasts between IPAD and the best-performing model (Linear regression) used to predict agricultural production for 2022 and 2023. For easier reference, this model is named Roux/UGB model in the rest of the paper. The model consistently outperforms IPAD in both years, achieving a lower MAPE.

2022 Forecasts: Forecasts were made in August 2022, near harvest time.

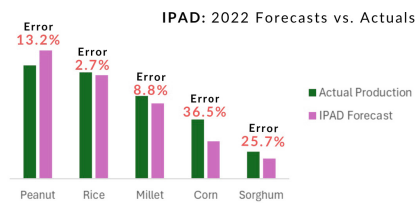


Fig. 6. IPAD Forecast 2022

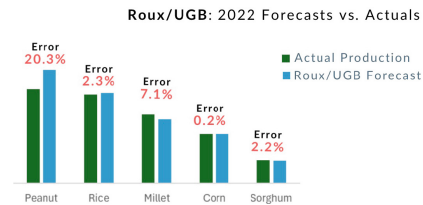


Fig. 7. Roux/UGB Forecast 2022

MAPE of Roux/UGB is 6.4% compared to IPAD's 17.4%, a 63% reduction in error rate.

2023 Forecasts: Forecasts were made in May 2023, earlier in the growing season.

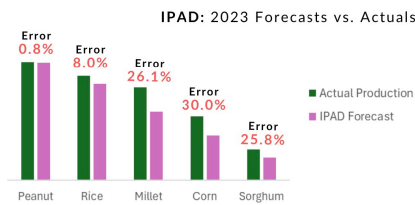


Fig. 8. IPAD Forecast 2023

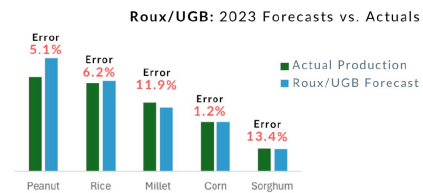


Fig. 9. Roux/UGB Forecast 2023

Roux/UGB model is 7.6% compared to IPAD's 18.1%, a 58% reduction in error rate.

This model shows more consistent performance across different crops and years, while IPAD's accuracy varies widely. This model could be improved for peanuts forecasts, especially in 2022. IPAD's model seems to struggle particularly with corn predictions in both years. 2023 forecasts were made in May 2023, earlier in the growing season, which might explain the slightly higher overall error rates for both models. Roux/UGB model demonstrates better adaptability to different crops and conditions, maintaining relatively low error rates across the board.

Finally, Roux/UGB model shows superior overall performance and consistency compared to IPAD's forecasts. It significantly reduces error rates by around 60% in both years, demonstrating its potential as a more reliable tool for agricultural production forecasts. The model's ability to maintain accuracy across different crops and forecast timings suggests it could be a valuable asset for agricultural planning and policy-making.

2024 Crop Production Forecasts. Building on successful forecasting model that has consistently outperformed IPAD predictions in recent years, Fig. 10 shows projections made for 2024.

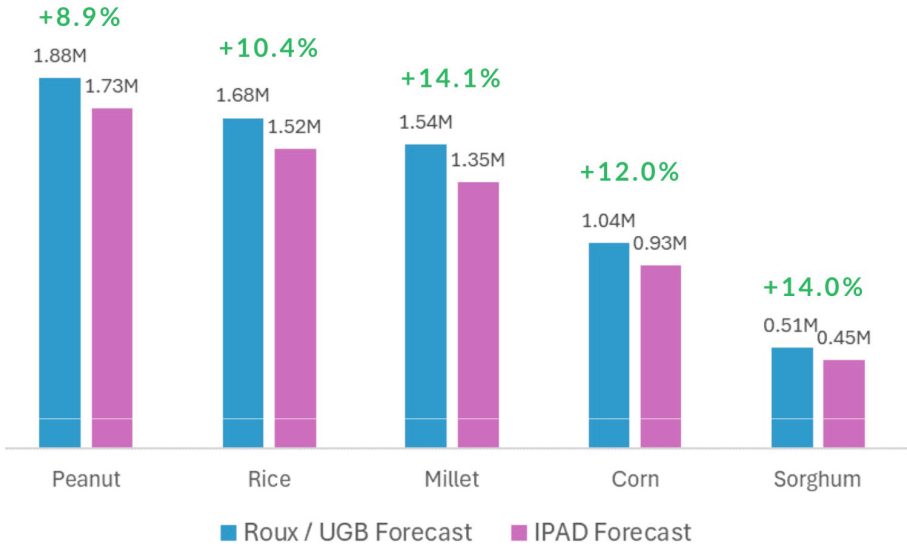


Fig. 10. 2024 Crop Production Forecasts (kg): Roux/UGB vs. IPAD

The production projections for various crops reveal the strengths and limitations of the model compared to IPAD estimates, offering valuable insights into the forecast accuracy and reliability.

For rice, the model projects a production of 1.68 million tons, representing a substantial increase of 10.4% compared to the previous estimates. This forecast significantly exceeds IPAD's estimate of 1.52 million tons, continuing the trend where the model predicts more optimistic and accurate productions for rice. The higher projection might be attributed to the model's ability to incorporate factors such as recent climatic improvements suggesting increased productivity in rice farming.

In the case of millet, a production of 1.54 million tons is anticipated, marking a 14.1% increase, which is notably higher than IPAD's 1.35 million tons forecast. This reinforces the model's consistent strength in millet production predictions over the past two years. The model's focus on drought-resistant crops could explain this continued superiority, as millet often grows in arid regions.

For corn, the projection stands at 1.04 million tons, reflecting a 12.0% increase. While this forecast is more conservative compared to other crops, it still surpasses IPAD's estimate of 0.93 million tons. The model has shown consistently high accuracy in corn production predictions, likely due to the integration of key variables like vegetation indices that capture factors crucial for corn cultivation.

The sorghum forecast of 0.51 million tons shows a 14.0% increase, once again outpacing IPAD's estimate of 0.45 million tons. The higher projection aligns with the model's trend of delivering more accurate sorghum production forecasts, pos-

sibly owing to its sophisticated time-series techniques, which effectively account for seasonal and environmental conditions impacting sorghum growth.

However, the model's limitations are evident in the case of peanuts, where IPAD's estimates have historically been more accurate. Trusting IPAD's 1.73 million kg projection for peanuts production reflects an acknowledgment of the model's underperformance in this area. This might suggest that the model struggles to capture certain factors unique to peanuts cultivation, that IPAD's methods may account for more effectively.

Overall, these projections highlight the model's strengths in forecasting productions for crops like rice, millet, corn, and sorghum, while acknowledging its limitations with peanuts, offering opportunities for further refinement.

The model is designed to allow reforecasting as new monthly data becomes available (e.g., reforecasting in September based on August rainfall). These projections indicate a significantly more productive agricultural year in 2024 compared to previous years.

This study presents a novel approach to crop production forecasting, demonstrating significant improvements over existing models, particularly those used by the IPAD. The results showcase the potential of sophisticated machine learning techniques in agricultural prediction, with implications for food security, economic planning, and climate change adaptation strategies.

The model demonstrates significantly better performance, as evidenced by the substantial reduction in error rates 63% in 2022 and 58% in 2023 when compared to IPAD forecasts. This reduction rate, defined as the percentage decrease in prediction error relative to the reference model (in this case, IPAD), highlights the effectiveness of the approach in improving forecast accuracy over consecutive years. By minimizing errors, the model delivers more reliable predictions for agricultural outcomes, surpassing IPAD methods.

This improvement can be attributed to a sophisticated methodology involving several key elements. First, a 20-year sliding window was employed for training data, enabling the model to capture long-term trends while remaining responsive to recent changes. Additionally, advanced time series analysis techniques, such as differencing, were applied to address the inherent seasonality and trends in agricultural data. The integration of diverse data sources, including climate data, vegetation indices (NDVI), and historical crop production data, provided a comprehensive understanding of factors influencing crop productions. Furthermore, the model underwent rigorous cross-validation using a walk-forward approach, ensuring robustness and generalizability.

The model's performance varies across different crops, offering insights into the complexity of agricultural forecasting. For corn, the model achieved remarkable accuracy, with error rates as low as 0.2% in 2022, compared to IPAD's 36.5%, suggesting that this approach is particularly well-suited for corn production predictions (see Fig. 6 and Fig. 7). In the case of peanuts, the model initially underperformed against IPAD in 2022 but showed significant improvement in 2023, indicating the ongoing refinement of the methodology. Consistent outperformance was observed for millet and sorghum, reflecting the model's strength

in predicting productions for more drought-resistant, locally significant crops. Finally, for rice, the model demonstrated comparable performance to IPAD, underscoring its robustness across various crop types, including water-intensive crops like rice.

Future Work. While the model demonstrates promising results, several areas require further exploration. One important direction is the integration of additional data sources, such as high-resolution satellite imagery, soil quality information, and pest prevalence data, which could significantly improve prediction accuracy. Additionally, testing the model's scalability and generalizability across various geographic regions and a wider range of crops is crucial to verify its broader applicability. Another critical aspect is the development of mechanisms for incorporating real-time data during the growing season, which would enhance the model's ability to make short-term forecast adjustments. Furthermore, expanding the model to project crop productions under different climate change scenarios could offer valuable insights for long-term agricultural planning.

As efforts continue to refine and enhance this model, its potential impact on agricultural planning, food security, and climate change adaptation strategies is expected to grow. Future research should prioritize improving model interpretability, increasing its geographical and crop coverage, and evaluating its usefulness for assessing the effects of climate change over time. Finally, the analysis underscores the importance of testing a variety of models and preprocessing techniques in agricultural forecasting, as simpler models can sometimes outperform more complex ones, depending on the data characteristics and the specific prediction task at hand.

5 Conclusion

In this study, the main idea was to develop a decision-support tool for decision-makers (government) and farmers to promote food security in Senegal. The potential of climate data, specifically rainfall and temperature, along with vegetation data, were evaluated to predict crop productions for rice, peanuts, corn, millet and sorghum in Senegal. To build the crop production forecast model, both traditional statistical methods and more advanced approaches such as machine learning were used with data aggregated over several years.

The study involved merging data on climate, vegetation, and agricultural production, followed by data cleaning and treatment of missing values. After pre-processing, which included feature engineering, an exploratory analysis was conducted to examine correlations between variables. Subsequently, the variables were transformed and encoded, and the data were normalized in preparation for model training. This process aimed to extract hidden information and make the data suitable for analysis. To predict annual crop productions, various machine learning models were compared, including Linear Regression, Ridge Regression, LASSO Regression, Random Forest, a baseline model, and ARIMA. During testing, some models did not deliver satisfactory results. Consequently, optimization

was performed by adjusting model parameters using cross-validation. The evaluation revealed that the Linear Regression model performed the best, achieving a Mean Absolute Percentage Error (MAPE) of 11.84%. The research highlighted the potential of the data used (climate, vegetation, production) to effectively make productions forecasts of the major crops under study. Thus, this study demonstrates the ability of Machine Learning techniques, combined with climate data, to revolutionize agricultural production forecasting in Senegal. This innovative approach offers a powerful tool for strengthening food security, optimizing agricultural policies and fostering the sustainable development of the agricultural sector in the face of the challenges of climate change.

The study faced several limitations, including limited data availability and instances of missing data, which may explain the model's underperformance in certain cases, such as peanuts productions forecasts (predictions for the year 2023). Additionally, the data granularity was restricted, as it was available only at an annual level for the entire country, resulting in fewer than 100 data points for analysis. Despite these constraints, the approach remains experimental and could be expanded by incorporating additional features such as soil data, wind data, humidity levels, and agricultural water data, among others. The incorporation of these additional features could enhance the accuracy of the forecasts. These forecasts, along with their integration, have the potential to enhance Senegal's resilience to the impacts of climate change on agriculture and contribute significantly to improving food security. By providing more accurate predictions, this study can help organisations that support sustainable agricultural practices and reinforce efforts to ensure food security in the face of climate-related challenges.

Acknowledgements. This research was supported by the United States Department of State (DoS): Embassy in Dakar, under the University Partnership Initiative (UPI), Grant No. SSG20023GR0032.

References

1. Faye, A., Dièye, M., Diakhaté, P.B., Beye, A., Sall, M., Diop, M.: Senegal - Land, Climate, Energy, Agriculture and Development: A Study in the Sudano-Sahel Initiative for Regional Development, Jobs, and Food Security, Jan. 19, 2021, Rochester, NY: 3769136 (2021). <https://papers.ssrn.com/abstract=3769136>. Accessed 16 Sept 2024
2. Toure, M.: Agricultural dynamics and food security in Senegal. *Int. J. Innov. Educ. Res.* **7**, 3 (2019). <https://doi.org/10.31686/ijier.vol7.iss3.1354>
3. Sarr, A.B., Sultan, B.: Predicting crop yields in Senegal using machine learning methods. *Int. J. Climatol.* **43**(4), 1817–1838 (2023). <https://doi.org/10.1002/joc.7947>

4. Zelingher, R., Makowski, D.: Investigating and forecasting the impact of crop production shocks on global commodity prices. *Environ. Res. Lett.* **19**(1), 014026 (2023). <https://doi.org/10.1088/1748-9326/ad0dda>
5. Zagre, I., Akinseye, F.M., Worou, O.N., Kone, M., Faye, A.: Climate change adaptation strategies among smallholder farmers in Senegal's semi-arid zone: role of socio-economic factors and institutional supports. *Front. Clim.* **6** (2024). <https://doi.org/10.3389/fclim.2024.1332196>
6. Yessoufou, A.N.-D., Kumar, S., Houessionon, P., Worou, O.N., Wane, A., Whitbread, A.: Vulnerability and resilience in the face of climate changes in Senegal's drylands: measurement at the household level and determinant assessment. *Front. Clim.* **6** (2024). <https://doi.org/10.3389/fclim.2024.1330025>
7. Altieri, M.A.: Agroecology: the science of natural resource management for poor farmers in marginal environments. *Agric. Ecosyst. Environ.* **93**(1), 1–24 (2002). [https://doi.org/10.1016/S0167-8809\(02\)00085-3](https://doi.org/10.1016/S0167-8809(02)00085-3)
8. Kamilaris, A., Kartakoullis, A., Prenafeta-Boldú, F.X.: A review on the practice of big data analysis in agriculture. *Comput. Electron. Agric.* **143**, 23–37 (2017). <https://doi.org/10.1016/j.compag.2017.09.037>
9. Pargaïen, S., Prakash, R., Dubey, V.P., Singh, D.: Crop cover identification based on different vegetation indices by using machine learning algorithms. *Environ. Res. Technol.* **7**(3), 3 (2024). <https://doi.org/10.35208/ert.1446909>
10. Wei, Z., Fang, W.: UV-NDVI for real-time crop health monitoring in vertical farms. *Smart Agric. Technol.* **8**, 100462 (2024). <https://doi.org/10.1016/j.atech.2024.100462>
11. Nyasulu, C., Diattara, A., Traore, A., Deme, A., Ba, C.: Towards resilient agriculture to hostile climate change in the Sahel region: a case study of machine learning-based weather prediction in Senegal. *Agriculture* **12** (9), 9 (2022). <https://doi.org/10.3390/agriculture12091473>
12. Wang, A.X., Tran, C., Desai, N., Lobell, D., Ermon, S.: Deep transfer learning for crop yield prediction with remote sensing data. In: *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS '18)*, New York, NY, USA: Association for Computing Machinery, pp. 1–5 (2018), <https://doi.org/10.1145/3209811.3212707>
13. Ibañez, S.C., Monterola, C.P.: A global forecasting approach to large-scale crop production prediction with time series transformers. *Agriculture* **13**(9), 9 (2023). <https://doi.org/10.3390/agriculture13091855>
14. Ansarifar, J., Wang, L., Archontoulis, S.V.: An interaction regression model for crop yield prediction. *Sci. Rep.* **11**(1), 17754 (2021). <https://doi.org/10.1038/s41598-021-97221-7>
15. Ujjainia, S., Gautam, P., Suraparaju, V.: Crop yield prediction using regression model. *Int. J. Innov. Technol. Explor. Eng.* **9**, 269–273 (2020). <https://doi.org/10.35940/ijitee.J7491.0891020>
16. Yasmin, S., Moniruzzaman, Md.: Forecasting of area, production, and yield of jute in Bangladesh using Box-Jenkins ARIMA model. *J. Agric. Food Res.* **16**, 101203 (2024). <https://doi.org/10.1016/j.jafr.2024.101203>
17. Hajaiah, D., Mathur, P., Jain, S., Nijhawan, S.: Crop yield prediction using machine learning and deep learning techniques. *Procedia Comput. Sci.* **218**, 406–417 (2023). <https://doi.org/10.1016/j.procs.2023.01.023>
18. Jorvekar, P.P., Wagh, S.K., Prasad, J.R.: Predictive modeling of crop yields: a comparative analysis of regression techniques for agricultural yield prediction. *Agric. Eng. Int. CIGR J.* **26**, 2 (2024). <https://cigrjournal.org/index.php/Ejournal/article/view/9247>

19. Patrick, S., Mirau, S., Mbalawata, I., Leo, J.: Time series and ensemble models to forecast banana crop yield in Tanzania, considering the effects of climate change, resources. *Environ. Sustain.* **14**, 100138 (2023). <https://doi.org/10.1016/j.resenv.2023.100138>
20. Attri, I., Awasthi, L.K., Sharma, T.P.: Machine learning in agriculture: a review of crop management applications. *Multimed. Tools Appl.* **83**(5), 12875–12915 (2024). <https://doi.org/10.1007/s11042-023-16105-2>
21. Elbasi, E., et al.: Crop prediction model using machine learning algorithms. *Appl. Sci.* **13**(16), 16 (2023). <https://doi.org/10.3390/app13169288>
22. Osibo, B.K., et al.: Interpretable deep learning model for crop yield prediction: a case study of wheat yield prediction in Egypt (2023). <https://doi.org/10.21203/rs.3.rs-3020861/v1>
23. Kumar, A., et al.: Comparative analysis of crop yield prediction using machine learning. In: 2023 International Conference on Advancement in Computation & Computer Technologies (InCACCT), pp. 310–315 (2023). <https://doi.org/10.1109/InCACCT57535.2023.10141745>
24. Ao, S.-I., Fayek, H.: Continual deep learning for time series modeling. *Sensors* **23**(16) (2023). <https://doi.org/10.3390/s23167167>
25. Gafurov, A., Mukharamova, S., Saveliev, A., Yermolaev, O.: Advancing agricultural crop recognition: the application of LSTM networks and spatial generalization in satellite data analysis. *Agriculture* **13**(9) (2023). <https://doi.org/10.3390/agriculture13091672>
26. Elavarasan, D., Durai Raj Vincent, P.M.: Fuzzy deep learning-based crop yield prediction model for sustainable agronomical frameworks. *Neural Comput. Appl.* **33**(20), 13205–13224 (2021). <https://doi.org/10.1007/s00521-021-05950-7>
27. Banda, E., et al.: Millet yield estimations in Senegal: unveiling the power of regional water stress analysis and advanced predictive modeling. *Agric. Water Manage.* **291**, 108618 (2024). <https://doi.org/10.1016/j.agwat.2023.108618>
28. Temizel, A., Halici, T., Logoglu, B., Temizel, T.T., Omruuzun, F., Karaman, E.: Experiences on image and video processing with CUDA and OpenCL. In: *GPU Computing Gems Emerald Edition*, pp. 547–567. Elsevier (2011). <https://doi.org/10.1016/B978-0-12-384988-5.00034-6>
29. Kulyal, M., Saxena, P.: Machine learning approaches for crop yield prediction: a review. In: 2022 7th International Conference on Computing, Communication and Security (ICCCS), Seoul, Republic of Korea, pp. 1–7. IEEE (2022). <https://doi.org/10.1109/ICCCS55188.2022.10079240>



Road Obstacle Detection Using YOLO Algorithm Based on Attention Mechanism

Bafokeng Lekola¹(✉)  and Serestina Lekola^{1,2} 

¹ School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal, Durban, South Africa

223153061@stu.ukzn.ac.za, viriris@ukzn.ac.za

² Centre for Augmented Intelligence and Data Science, University of South Africa, Pretoria, South Africa

Abstract. Road obstacle detection is an important task in autonomous vehicles (AVs) and advanced driver assistance systems (ADAS) as they require real-time operation and high accuracy for safe operation. The mobile nature of the task means that it is carried out in a low-resourced environment where there is a need for an algorithm that achieves both high accuracy and high inference speed while meeting the requirement for lightweight. In this paper, an exploration of the effectiveness of the attention-enhanced YOLO algorithm for road obstacle detection is carried out. Several state-of-the-art attention modules that employ both channel and spatial attention are explored and fused with the YOLOv8 and YOLOv9 algorithms. These enhance feature maps of the network by suppressing non-distinctive features allowing the network to learn from highly distinctive features. The Attention-modified networks are trained and validated on the Kitti and BDD100k datasets which are publicly available. Comparisons are made between the models and the baseline. An improvement from the baseline is seen with the GAM attention achieving an mAP_{50} of 93.3 on the Kitti dataset and 71.1 on the BDD100k dataset. The Attention modules generally achieved incremental improvements over the baseline.

Keywords: Road Obstacle · Obstacle Detection · YOLO Algorithm · Attention Network

1 Introduction

Changes in the current technology landscape have seen a significant increase in computation capabilities brought about by advancements in technology. One such technology is cloud computing, which has brought forth exponential growth in the AI space by providing computing resources once unavailable to many people. This has led to the emergence of multiple breakthrough technologies. We have seen state-of-the-art performances from DL and ML fields, such as

the usage of GANs (Generative Adversarial Networks) in computer vision and LLMs (Large Language Models) in NLP, bringing attention to the AI field. These models have been able to achieve state-of-the-art results by leveraging the high computing capabilities offered by expensive cloud service offerings. One of the technologies that has seen a rise is that of advanced driver assistance systems (ADAS) and autonomous vehicles which, coupled with the advancements in perception technology, compute capabilities, machine learning and deep learning have constituted a popular research area due to their potential to improve road safety and reduce accidents. One of the critical tasks for ADAS and autonomous vehicles is obstacle detection on the road. Accurate and timely detection of obstacles on the road is essential for ensuring the safety of passengers and other road users. The usage of onboard sensors like cameras and lidar is very important in this regard and allows for the necessary data collection for obstacle detection [41].

Object detection is a crucial task for autonomous vehicles and ADAS systems, and thus we seek to find a balance between real-time object detection and accurate detection. This is because the system running on the vehicle has access to limited resource capabilities in power and computing but also requires high accuracy for the sake of safe operation [10].

Multiple models have been proposed for this task which seek to achieve this trade-off balance. These models have had a problem with road obstacle detection for small obstacles. Among the various object detection algorithms, single-stage algorithms have been seen to be the best for real-time detection and particularly the You Only Look Once (YOLO) algorithm has emerged as the most popular method due to its speed and accuracy. However, the performance of the YOLO algorithm can be limited in detecting small or occluded objects, which are common in real-world scenarios, such as road obstacle detection [40].

This paper proposes a YOLO algorithm based on an attention mechanism that can accurately detect small and occluded obstacles on the road in real-time. The proposed algorithm is evaluated using benchmark datasets for road obstacle detection, and compared with other state-of-the-art object detection algorithms. The research provides insights into the impact of the attention mechanism on the detection performance and processing speed of the YOLO algorithm and the algorithm's robustness in different lighting and weather conditions. The contributions of this paper are:

- Addition of Attention mechanisms into the neck region of the YOLOv8 model and training for road obstacle detection.
- Benchmarking the modified YOLOv8 algorithms on the Kitti and the BDD100k datasets.
- Evaluation of the performance of the Attention Network-based YOLO framework against other state-of-the-art models in the literature.

2 Related Work

2.1 Autonomous Vehicles and Advanced Driver Assistance Systems

It is hard to imagine life without the automobiles that we rely on daily. This allows us to travel distances long and short and also allows the transportation of goods just as easily. The domain of autonomous vehicles has seen a huge boost in this recent surge in technological advancement such that multiple automobile manufacturers investing billions in R&D in this space. With the usage of hardware like cameras, lidars and multiple sensor technologies, data is captured and processed for use in assistive tasks such as collision avoidance [35], pedestrian detection [8], traffic sign recognition [34] lane departure [44], etc. thus improving the elimination of errors that may be caused by human incompetence/shortcomings [6].

Assistive driver assistance technologies and thus autonomous vehicles have improved over the years is shown in Table 1 below. Achievement of such results means that the data processing should be done to the best possible standard to avoid causing more harm than what was intended by the adoption of the technology in the first place. This is why deep learning techniques have been used in the data processing part as they produce the best results.

Table 1. Different levels of vehicle automation [22]

Levels	Features
0	Fully driven. No automation at all
1	Driver assistance. cruise control, automatic braking, and lane keeping are included
2	Driver required to observe. Some automated functions like steering assist, automatic parking, collision braking are included
3	Conditionally automated. The vehicle can self-drive with driver observing the environment but the driver can take control at any time
4	High automation. Under certain conditions, the vehicle can self operate
5	Fully automated. The vehicle can self drive fully under any condition

2.2 Object Detection Methods

Early research in the field of object detection used multi-stage methods of detection that included the proposal of regions in an image for processing, feature extraction from the proposed regions and then finally classification. These are seen in the first stages of Fig. 1 which shows the progression of object detection algorithms over time. The most widely used of the feature extraction algorithms include the Scale Invariant Feature Transform (SIFT) [20], Local Binary Patterns

(LBP) [24], Histogram of Oriented Gradients (HOG) [5] and the Haar wavelets [17]. These algorithms rely on identifying edges, key points and pattern matching in order to identify features present in image. These features are then passed through a classifier such as the support vector machine (SVM) [12]. The multiple stages included in this methods and the number of operations performed means that this methods take a long time to process information and will require a lot of computing capabilities.

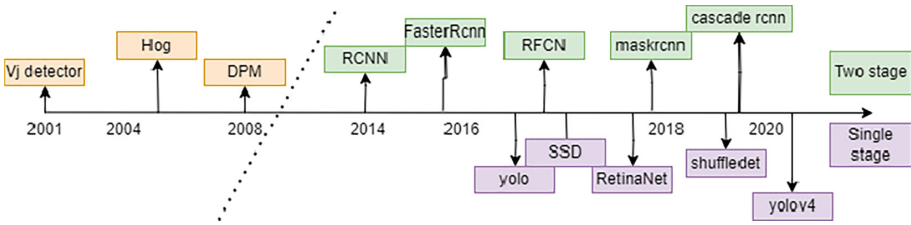


Fig. 1. The progression of object detection algorithms [22]

Most recently as shown in the latter part of Fig. 1 with the arrival of graphical processing units (GPUs) and large cloud computing platforms, deep learning research saw an exponential rise and with it a rise in deep learning techniques used for object detection. Deep learning methods have proven to be more efficient for image processing tasks such as object detection, lane detection, image segmentation, etc., than the traditional methods.

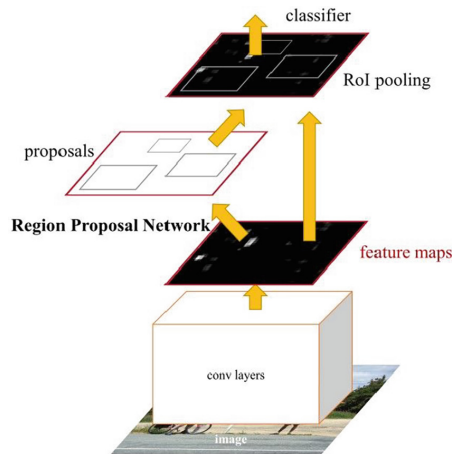
These methods utilize the CNN architecture for feature extraction instead of using the traditional methods. These usually have a predefined convolutional network as the backbone of the network that performs the feature extraction, with classification and localization of the extracted features done in the final stages of the network. Some of the CNNs commonly used as network backbones are as seen in Table 2. These networks usually have varying parameter sizes and network depth which directly influence their performance in feature extraction. Several algorithms have emerged in the field of object detection that utilize these backbone networks and have been seen to achieve state-of-the-art results. These are classified into two categories mainly the single-stage detectors and the two-stage detectors. With two-stage algorithms performing proposal of regions of interest first and then detection after, the single-stage algorithms are seen to be more efficient and perform faster as they only employ a single pass of the network to perform detection although with lower accuracy.

The *two-stage* algorithms include the Faster-RCNN [30] algorithm that builds on top of the fast RCNN [9] algorithm by sharing the convolutional layers for the two regions: the Region Proposal Network(RPN) and the Fast-RCNN detector as can be seen by the separation in the two paths after the conv layers in Fig. 2 and the RFCN algorithm that Builds on the Faster RCNN method by introducing position-sensitive score maps using the convolutional neural network.

Table 2. Popular Backbone/feature extraction networks

Model	#Parameter(M)	ImageNet-1k Top-1 Accuracy (%)	Year
Alexnet [15]	62.36	63.32	2012
VGG [31]	138.37	73.01	2014
GoogLeNet [32]	25.56	76.13	2014
EfficientNetV2-S [33]	21.5	84.23	2021
ShuffleNetV2 [21]	7.4	76.23	2018
MobileNetV3-Large [26]	5.5	75.27	2021

This aims to address the problem that comes with the need for position sensitivity in object detection in an image and the translation invariance required for the classification of the image in the shared network. Figure 3 shows the architecture of the RFCN network showing the positioning of the backbone, position-sensitive score maps and the region proposal network regions.

**Fig. 2.** The faster RCNN network [30]

The *single-stage* algorithms include the SSD [19] that generates anchor boxes at different scales/ratios for the different size feature map locations and creates scores for each object category present in each default box during prediction time and modifies the box to better fit the object's shape. To handle objects of varied sizes naturally, the network additionally incorporates predictions from numerous feature maps with different resolutions. The SSD network uses the VGG 16 [31] network for feature extraction on top of which 6 more convolutional layers are added for detection. The YOLO algorithm is the most popular of the single stage algorithm and has seen mass adoption in object detection.

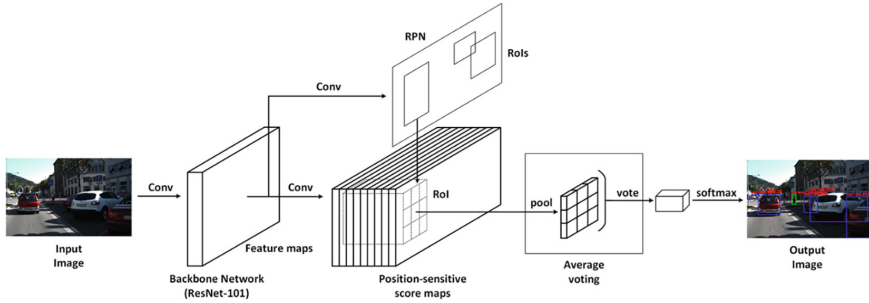


Fig. 3. R-FCN architecture [4]

The algorithm has seen multiple improvements since the original YOLO algorithm [27] that have seen the emergence of what we term the YOLO series of algorithms [1, 14, 16, 27, 29, 36, 39]. These algorithms are a progressive improvement and introduce multiple novel methods to increase the performance and efficiency of the original algorithm. For example, the YOLOv2 [29] algorithm introduces batch normalization, and high resolution and introduces the Darknet-19 backbone for feature extraction to name a few improvements. Other improvements include adding a spatial pyramid pooling network (SPP) [11] to gather features from multiple scales, allowing for a larger receptive field, introducing anchor boxes and then introducing an anchor-free approach in the latest YOLOv8 algorithm, different loss functions, etc. To improve the detection accuracy of algorithms, the addition of attention mechanisms has been proposed. To address the problem that single-stage algorithms mainly the YOLO algorithms have in detecting small and occluded objects this has provided promising results.

Attention mechanisms have been widely used in natural language processing and computer vision to selectively focus on relevant features or regions of interest. In object detection, attention mechanisms can help the model to selectively attend to the most informative features and suppress the irrelevant ones, which can improve the detection performance [2]. Several recent studies have explored the use of attention mechanisms in object detection, including in YOLO-based methods.

Several studies have applied attention mechanisms to the YOLO algorithm to improve its performance in detecting small objects. [13] proposed an attention-based YOLOv5 algorithm for multispectral pedestrian detection to provide robust and accurate detection under different illumination conditions, which achieved state-of-the-art performance on benchmark datasets. Similarly, Chen et al. [23] proposed an attention-guided Tiny-YOLOv4 algorithm for vehicle and pedestrian detection to solve the problem of rapidly changing detection scales due to a fast-moving vehicle, which also outperformed other state-of-the-art object detection algorithms. These studies and more show that attention is useful in improving the YOLO models.

Similarly, [3] proposed an attention-based YOLOv5 algorithm for traffic sign recognition. The algorithm used an attention mechanism to focus on the regions of interest and improve the detection accuracy of small traffic signs on the traffic sign dataset named TT100K. The study showed that the speed of the attention-based model had significant improvements in accuracy on the small target objects and the model can meet the requirements for a vehicle-mounted detection system.

Overall, these studies suggest that attention mechanisms can be effective in improving the performance of the YOLO algorithm for road obstacle detection.

3 Proposed Methodology

3.1 Network Structure of the YOLOv8

The YOLOv8 is one of the iterations of the YOLO algorithms which was presented in 2023 by Glen Jowher at Ultralytics whose structure is illustrated in Fig. 4. This algorithm has been proven to achieve better performance than all previous versions.

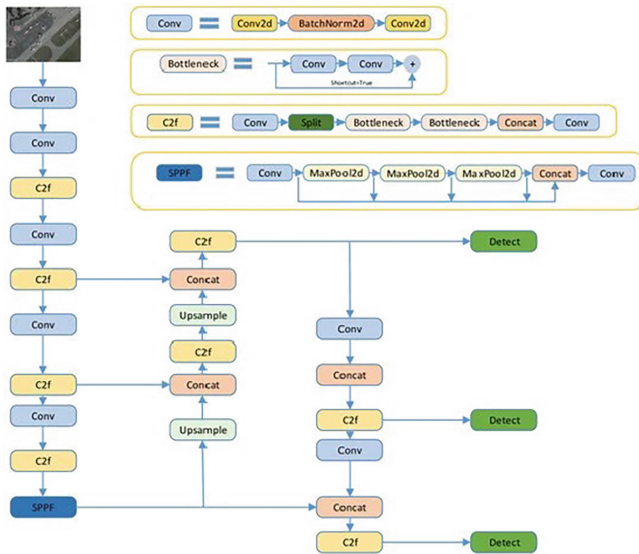


Fig. 4. The YOLOv8 architecture

The key modifications added to the YOLOv8 include:

CSPDarknet53, a Darknet architecture variation, as its feature extractor. SiLU activation functions, batch normalization, and convolutional layers make up this component. Notably, for better feature extraction, YOLOv8 substitutes a 3×3 convolutional layer for the original 6×6 convolutional layer.

Cross-Stage Partial Bottleneck (C2f) Module: YOLOv8 presents the C2f module, which efficiently combines contextual data with high-level features. The bottleneck blocks' outputs, which are two 3×3 convolutions with residual connections, are concatenated to accomplish this. The goal of this architectural modification is to improve feature representation and to balance between lightweight implementation and good performance. The inspiration for this module is drawn from the C3 module of the YOLOv5 architecture.

Spatial Pyramid Pooling Fast (SPPF), to capture features at multiple scales enhancing the performance of the algorithm at different object sizes.

Neck and Head A Path Aggregation Network (PANet) [18] is introduced as the neck structure of YOLOv8. By facilitating information flow across various spatial resolutions, PANet helps the model effectively capture features at several scales. By using an anchor-free detection approach, YOLOv8 predicts object centres without the need of predefined anchor boxes. It makes use of an altered YOLO head that has a new Intersection over Union (IoU) loss function and dynamic anchor assignment. These enhancements help handle overlapping items better and forecast bounding boxes with more accuracy. It comprises of multiple detection heads that make up the complete head structure that are responsible for projecting bounding boxes, class probabilities, and object ratings on various scales. The model comes in multiple sizes and depending on the speed to accuracy tradeoff that is acceptable, anyone of the models can be chosen. In this paper we chose to use the large model because of the comparison to be done with the YOLOv9 model which comes with the number of parameters equal to that of the YOLOv8l.

3.2 Network Structure of the YOLOv9

This algorithm addresses the non-negligible information loss encountered during the feedforward process which leads to diminished information being used to upgrade the weights of the network. This information loss is detrimental in the usage of deep CNN for small object detection as small objects occupy a small portion of the network and thus are very susceptible to being lost due to diminished gradients.

The Programable Gradient Information (PGI): In order to enhance the model's detection capacity, PGI provides an auxiliary supervision framework intended to control the transmission of gradient information across multiple semantic levels. The primary branch, auxiliary reversible branch, and multi-level auxiliary information make up PGI. The main branch manages both forward and back propagation during the inference phase. An information bottleneck could develop as the network gets deeper, resulting in loss functions that are unable to provide helpful gradients. Reversible functions are then used by the auxiliary branch to feed back the necessary information back into the main branch to protect information integrity and lessen information loss. Since the auxiliary branch is only used for the propagation of gradients, it can be removed after training and inference performed only by the main branch. The multi-level auxiliary information is then used to aggregate returned gradients from several

prediction heads by inserting an integration network between the main branch and the feature pyramid hierarchy tiers of auxiliary supervision to tackle the problem of error accumulation from the deep supervision mechanism, enhancing the model’s learning ability by adding additional information at different levels. The structure of the PGI is as depicted in Fig. 5, showing the structure of branches employed.

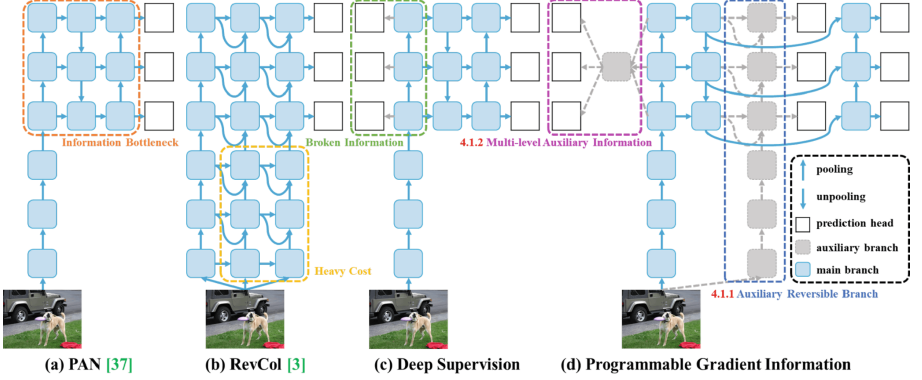


Fig. 5. Structure of the YOLOv9 PGI [39]

Generalized Efficient Layer Aggregation Network. Combines concepts from the CSPNet [37] architecture and the ELAN [38] which are designed for gradient path planning. Provides a lightweight architecture that helps the network to more efficiently use information across layers and helps the network to avoid the information bottleneck problem leading to increased accuracy and speed.

3.3 Proposed Framework

In this work, different implementations of attention mechanisms are incorporated into the neck region of the two YOLO models, v8 and v9. This is to enhance the features extracted from the lower layers of the network, making them ripe for use in the detection head parts of the network. The proposed schematic of the proposed system is in Fig. 6. The red sections represent the location in which the GAM attention, ECA attention, CBAM attention and simAM attention will be added. Since the attention module performs feature enhancement, these areas should benefit and hence the network should perform better.

4 Datasets and Performance Metrics

4.1 Datasets

We utilize two of the most popular publicly available datasets for this research in Kitti [7] which consists of 14999 images with half of them with image anno-

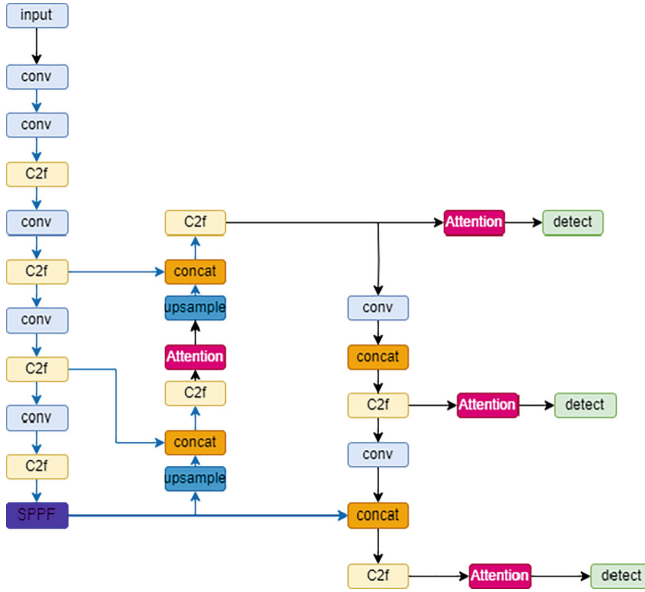


Fig. 6. Proposed YOLOv8 with Attention Mechanism

tations. The labelled images comprise 51865 object annotations. We merge the original classes into just six classes down from nine allowing for the reduction in complexity and better generalization capabilities of the models. This also allows for the comparison with those in the literature. These images included many different driving and traffic scenarios, driving in the city, highways, and rural areas. For this research, we use the labelled part of the dataset and split that into training, validation and test sets using the usual 7:2:1 ratio.

The BDD100k [43] dataset which has 2221128 bounding box annotations across 10 different categories on 100k images sampled from multiple crowd-sourced image sequences. The dataset includes data from different driving scenes including city streets, highways and residential areas across multiple day and night weather conditions. This diversity gives the dataset an advantage over other popular datasets allowing for the evaluation of different domains for object detection. The dataset is split three ways into training(70000), validation(10000) and testing(20000). We merge and drop some of the categories from the dataset to form just the four categories (vehicle, pedestrian, traffic sign, traffic light). This is to allow for a proper comparison with literature as will be seen in the interpretation of results below. The difference between the two datasets mainly being the number of images and differing weather conditions found in the BDD100k dataset.

4.2 Evaluation Metrics

In this research, mean average precision (mAP) is used to evaluate the model's performance. This is the most widely used method for evaluating object detection models and provides a very good measure of model performance across multiple classes allowing for a detailed analysis. The *mAP* is calculated by using two other metrics which are the precision (P) and recall (R) and is obtained by computing the area under the precision/recall curve.

Precision is defined as the percentage of samples detected as correctly positive. It is calculated by Eq. 1. At the same time, recall is the percentage of samples detected as correctly positive for all the ground-truth samples and is calculated by Eq. 2. The parameters for this calculation are False positives (FP), false negatives (FN), true positives (TP) and true negatives (TN).

True positives are the correct detection, false positives are the samples detected as correct but are not, false negatives are samples detected as false but are correct and true negatives are correctly detected as false.

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

The TPs are computed at object bounding box intersection-over-union (IOU) of 0.5. The usage of these two metrics is ideal for object detection as they focus only on positive detection negating the true negatives which may overwhelmingly affect the detection performance as there may be infinitely many TN [25]. The calculation for average precision (AP) is given by the equation 3 which computes the area under the precision/recall curve mAP will then be the averaging of the APs for the different classes given by equation 4.

$$AP = \int_0^1 P(R) dR \quad (3)$$

$$mAP = \frac{\sum_{i=1}^M AP_i}{M} \quad (4)$$

5 Experimental Results and Analysis

5.1 Experimental Environment and Training Parameters

Experimentation was done using the Center for High Performance Computing cluster, South Africa environment of Tesla V100 16GB GPUS accessed individually for training. This was done in order to utilize the high compute capabilities offered for research by the center. All code was implemented on pytorch using the ultralytics code base [14]. Table 3 shows the experimental and hyperparameters settings.

Table 3. Experimental Settings and Hyperparameters

Setting	Hyperparameters
Epochs	190
Optimizer	SGD with Momentum
Learning Rate	0.01
Mometum	0.937
LR Scheduler	11, 15
LR Decay	0.0005
Weight Decay	0.001
Batch Size	16
NMS: IOU Threshold	0.7
Resolution	640×640
Weight Initialization	from scratch

5.2 Experimental Results on the Kitti Dataset

This section presents the results of the different models on the Kitti dataset.

Comparisons of Models Across mAP Values. In this section, the models are compared based on the overall performance across all class categories. The models are trained on 6 classes of the kitti dataset. We compare using the 4 metrics precision, recall, mAP_{50} and the mAP_{50-95} . The mAP_{50} and the mAP_{50-95} are the mAP at bounding box IOU threshold of 0.5 and average of the mean average precision calculated at varying IoU thresholds, ranging from 0.5 to 0.95 respectively. For the YOLOv8 variants, the attention modules were added to the neck region in the sections after the C2f module. There are four attention modules added to the neck section of the yolov8 network. Table 4 shows that the GAM attention achieves the best performance increase amongst all the attention implementations of 1%. It can be seen that there is an increment in both precision metrics and mAP_{50} across the models but not so much on the recall metric.

Table 4. Results of modified YOLOv8 models on the kitti dataset

Yolov8 model	Precision	Recall	mAP_{50}	mAP_{50-95}
Original	91.1	86.8	92.3	71.9
ECA	92	86.4	92.8	71.4
simAM	92.8	86.3	92.7	71.3
GAM	93.6	82.7	93.3	72.6
CBAM	92.3	87.4	92.7	70.8

Comparison of Models Across Categories. This section compares the performance of models across the different class categories. Table 5 shows that, across categories, the GAM attention network sees an improvement over multiple categories hence the greater improvement in overall mAP_{50} value. The car category is the most prominent one as with almost if not all autonomous driving datasets having a class imbalance that favours it, it can be seen that all the variants present the same value which has very little room for improvement.

Table 5. Results of modified YOLOv8 models on the kitti dataset categories

Class	Original	ECA	CBAM	simAM	GAM
Car	98	98	98	98	98
Cyclist	85.4	86.1	85.5	86.2	86.3
Pedestrian	85.2	86.4	85.6	85.2	86.2
Tram	94.5	95	94.8	94.8	96.8
Truck	94	94.3	95.2	94.9	95.5
Van	96.9	96.7	97.2	97.1	97.3

5.3 Results on the BDD100k Dataset

In this section results on the BDD100k dataset are presented. Experiments on the datasets are done using modified YOLOv8 models of different sizes on the BDD100k dataset.

Comparisons of Models Across mAP Values. In this section, the model comparisons are made based on the overall performance across all class categories. We compare using the 4 metrics precision, recall, mAP_{50} and the $mAP_{50} - 95$. Table 6 below presents performance results on the yolov8 variations on the BDD100k dataset. The results show that there are some performance increments on some of the models such as the nano and small versions of the CBAM and the GAM attention models. This could be because training on an extended dataset such as the BDD100k dataset provides a large enough sample size such that an advanced model such as the YOLOv8 is able to learn as much as possible. The attention models that produce performance increases also have a notable increase in parameter count over the baseline network which could be the reason why they give a performance boost. An increase in parameter count and complexity can also be seen in the increasing performance metrics as the model sizes increase though also at the tradeoff in inference speed of the networks. YOLOv9-tiny and small baseline models are trained in the same manner as the YOLOv8 models to give a fair comparison between models that have the same parameter count. It can be seen from the results that, even though the YOLOv9 algorithm is the latest between the two and also draws inspiration from the YOLOv8 algorithm does not perform better. The YOLOv9 does not perform as well as the YOLOv8 algorithm even though it trains twice as long.

Table 6. Results of modified YOLOv8n and YOLOV8s models on the BDD 100k dataset

Yolov8 model	Precision	Recall	mAP_{50}	mAP_{50-95}
Original(n)	68.7	53.6	59	28.7
Original(s)	72.8	58.7	64.7	32.1
Original(l)	75.1	63.9	69.9	35.9
ECA(n)	68.7	53.7	59	28.7
ECA(s)	72.1	58.7	64.6	32.1
ECA(l)	75.7	63.9	70	36
simAM(n)	68.2	53.6	58.8	28.6
simAM(s)	72.1	58.4	64.4	32
simAM(l)	75.2	63.9	69.8	35.9
GAM(n)	69.8	54.4	60.3	29.5
GAM(s)	72.1	58.7	64.6	32.1
GAM(l)	75.8	65.1	71.1	36.5
CBAM(n)	69.3	54.4	59.9	29.3
CBAM(s)	72.8	58.9	65	32.4
CBAM(l)	74.8	62.3	68.6	34.9
YOLOv9-t	66.1	49.9	55.6	27.6
YOLO9-s	70.7	56.4	62.6	31.7

Comparison of Models Across Categories. Performance across object categories is presented and compared between the different models. The results presented in Table 7 for the models across categories show that the GAM attention and CBAM attention modules achieve a performance boost over the baseline more than the other attention mechanisms. This shows that attention has the potential for feature enhancement depending on the dataset used and the type of attention used.

5.4 Comparison with Literature

In this section, we compare the results of our algorithms with results found in the literature. The BDD100k dataset provides us with a sizable amount of literature to compare against since it has been widely used for this task. The models are compared on the same metrics as the YOLOP [42] which combines the four classes bus, car, truck, train into a single class “vehicle”. Table 8 presents the results of the vehicle class alone for comparison. Table 9 presents the comparison of proposed models with respect to literature. It can be seen that the results achieved are on par with the results achieved by state-of-the-art models. These models are trained as multitasking models hence the performance they achieve over single-task models like ours may be the result of training strategies that

Table 7. Results of modified YOLOv8n and YOLOV8s models on the BDD 100k dataset categories

Yolov8 model	Vehicle	Pedestrian	Traffic light	Traffic sign
Original(n)	75.8	51.4	51	57.8
Original(s)	78.9	58.5	57.3	64
Original(l)	81.7	65.4	62.8	69.7
ECA(n)	76	51.1	51.1	57.9
ECA(s)	79	58.2	57.5	64.2
ECA(l)	81.7	65.3	63.2	69.8
simAM(n)	75.8	51.2	50.8	57.5
simAM(s)	78.8	57.9	56.7	64.1
simAM(l)	81.7	65.2	62.8	69.5
GAM(n)	76.6	53.4	52.3	58.8
GAM(s)	78.8	58.5	57.2	64.2
GAM(l)	82.4	66.9	64.1	70.9
CBAM(n)	76.4	53.1	52	58.1
CBAM(s)	79.2	58.8	57.7	64.2
CBAM(l)	81.2	63.5	61.3	65.8
YOLOV9-t	74.9	47.2	45.3	54.8
YOLOV9-s	78.7	55.2	53.3	63.1

may have an implicit influence on the individual tasks [10]. For this comparison, the best-performing model from Table 8 is used for each model size.

5.5 Example Visualizations of Attention Heatmaps

The attention heatmap of the network helps to visualize the outputs of parts of the network and see the different parts of the image that assist in the determination of the final output detection. This also helps to understand the performance of the network model and how it operates. A close look at Fig. 7 shows that the attention models do have a big impact in the computed feature maps of the network. In comparison with the original YOLOv8, though most of the same parts of the image are targeted, the attention networks seem to have accentuated the focus region more and in some instances provided a focus on parts of the image that were not seen by the original network. In images c), d) and e), there's parts that have been isolated on the mid-left of the image that contain instances of vehicles not seen by the original network. d) Also identifies a traffic sign not seen by the other networks

Table 8. Results on the single class vehicle

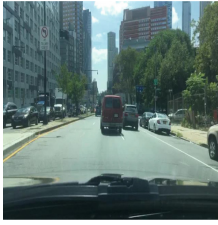
Yolov8 model	Precision	Recall	mAP_{50}	mAP_{50-95}
Original(n)	76.4	69.1	75.8	45.9
Original(s)	80.2	71.2	78.9	48.5
Original(l)	75.1	63.9	69.9	35.9
ECA(n)	76.1	69.3	76	46
ECA(s)	80	71.7	79	48.5
ECA(l)	75.7	63.9	70	36
simAM(n)	76.1	69.3	75.8	45.9
simAM(s)	79.4	71.8	78.8	48.4
simAM(l)	75.2	63.9	69.8	35.9
GAM(n)	77.2	69.8	76.6	46.5
GAM(s)	79.3	72	78.8	48.4
GAM(l)	82.9	75.6	82.4	51.7
CBAM(n)	77.1	69.8	76.4	46.5
CBAM(s)	80	72.1	79.2	48.8
CBAM(l)	74.8	62.3	68.6	34.9
YOLOv9-t	73.6	68.3	74.9	27.6
YOLOv9-s	78	71.4	78.7	49

Table 9. Comparing models with state-of-the-art models

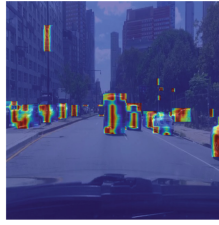
Model model	Recall	mAP_{50}	Parameters
HybridNets	92.8	77.3	12.8M
YOLOP	88.6	76.5	7.9M
YOLOPV2	91.1	83.4	38.9M
YOLOv8(n)	69.1	75.8	3M
A-YOLOM(n)	85.3	78.0	4.43M
A-YOLOM(s)	86.9	81.1	13.61M
YOLOv8(n)-GAM	69.8	76.6	3.69M
YOLOv8-CBAM(s)	72.1	79.2	16.05M
YOLOv8(l)-GAM	75.6	82.9	82.9M

5.6 Detection Comparison of Models

Detection results from the different networks are analysed visually in this section. It can be seen from Fig. 8 that the models have varied detection performance on the image and the attention models are more or less on par with the original YOLOv8, but have detected some instances not seen by the original. On the left side of the image, there are vehicle instances that have been picked up by the



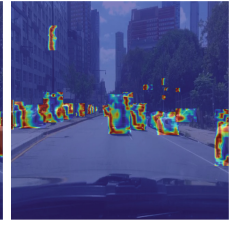
(a) image



(b) Original yolov8



(c) GAM attention



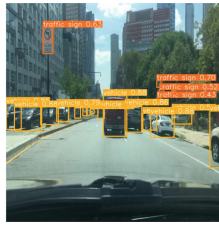
(d) CBAM attention



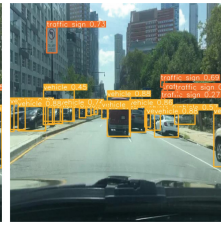
(e) ECA attention

Fig. 7. Attention Heatmaps in the daytime

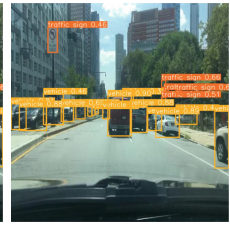
(a) image



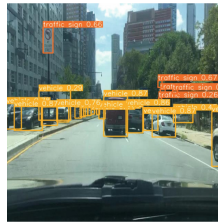
(b) Original yolov8



(c) GAM attention



(d) CBAM attention



(e) ECA attention

Fig. 8. Example image detections in the daytime

attention-enhanced models. It can also be seen that the original network sees the street light on the left as two instances whereas there is only one. These results though not overwhelmingly so, show that attention added to the YOLOv8 can provide some performance enhancements to the network.

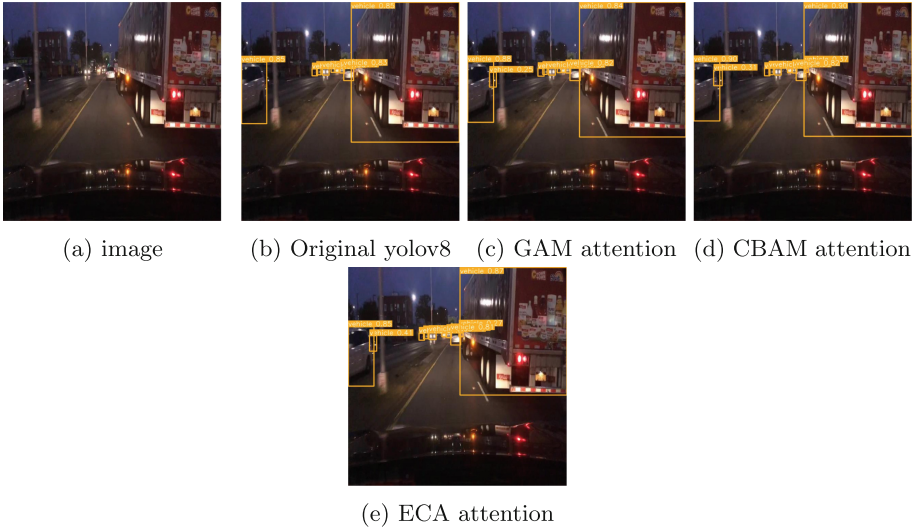


Fig. 9. Example image detections in the nighttime

5.7 Comparison of Detection Across Different Weather Conditions

Example prediction results are plotted and visual observations are made on the results of the different models. From these results, it can be seen that even though most of the detections made are similar for all the models, it can be seen that the attention models achieve higher confidence in detection than the original network. It can be seen in Fig. 9 that the GAM attention achieves a confidence score of 90% whereas the original YOLO achieves 85%. It can also be observed that the original YOLO also seems to miss some instance of vehicle on the left which can be detected by all the other models.

For Fig. 10, the models seem to be performing on par though the original network does detect a traffic sign not detected by the other models. The confidence scores of the models on the different objects is more or less the same. This shows that in the rain, the models perform equally. This could be the result of fewer samples on such conditions than normal daylight samples and also aligns with the difficulties that humans face with visibility in this conditions. It can be seen that the vehicle class is detected at a higher confidence score than the other classes. This could be caused by the area covered by the vehicle class in each image and the general class imbalance of the dataset.

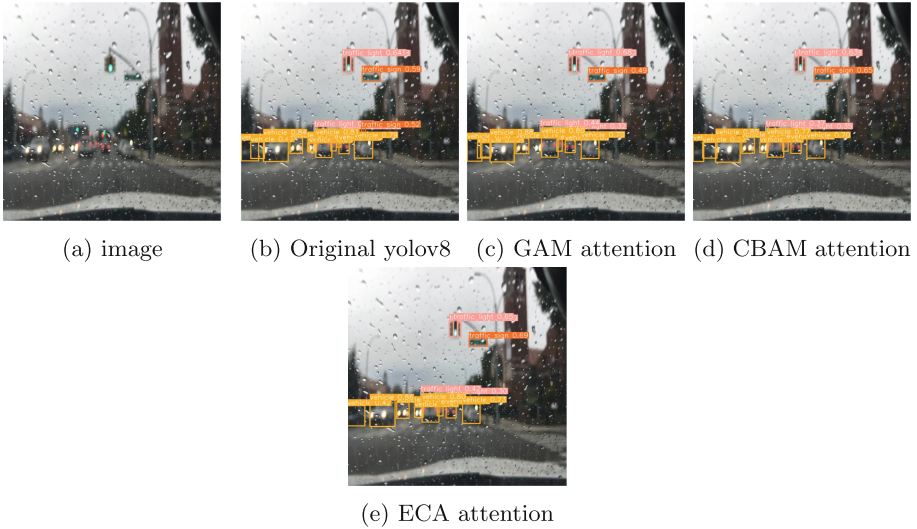


Fig. 10. Example image detections in the rain

6 Conclusion

The attention mechanisms have proven to be effective in improving neural network performances in multiple fields for object detection. This paper explores the effectiveness of the fusion of Attention Network with the latest YOLO algorithms for the task of road obstacle detection. This was done by incorporating Attention modules into parts of the YOLO models for the sake of feature enhancement. The networks were trained on two datasets, the Kitti dataset and the BDD100k dataset. From the attention modules that were implemented, the GAM attention module was found to be the best-performing model achieving mAP_{50} of 93.3% on the Kitti dataset and 71.1% on the BDD100k dataset. It was also seen that the YOLOv8 outperforms the YOLOv9 with the same number of parameters. The trained models were then compared with state-of-the-art models and were seen to perform on a level close to them. Visual inspection of the results showed that the attention-enhanced models were able to detect instances of occluded vehicles that were not detected by the original network. Some instances of traffic signs were also detected by the enhanced models but not the original. This shows that the attention was able to improve the detection performance of the network as expected.

References

1. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint [arXiv:2004.10934](https://arxiv.org/abs/2004.10934) (2020)
2. Chaudhari, S., Mithal, V., Polatkan, G., Ramanath, R.: An attentive survey of attention models. *ACM Trans. Intell. Syst. Technol. (TIST)* **12**(5), 1–32 (2021)
3. Chen, Y., Wang, J., Dong, Z., Yang, Y., Luo, Q., Gao, M.: An attention based yolov5 network for small traffic sign recognition. In: 2022 IEEE 31st International Symposium on Industrial Electronics (ISIE), pp. 1158–1164 (2022). <https://doi.org/10.1109/ISIE51582.2022.9831717>
4. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. *Adv. Neural Inform. Process. Syst.* **29** (2016)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 1, pp. 886–893. IEEE (2005)
6. Dilek, E., Dener, M.: Computer vision applications in intelligent transportation systems: a survey. *Sensors* **23**(6), 2938 (2023)
7. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: the kitti dataset. *Inter. J. Robot. Res.* **32**(11), 1231–1237 (2013)
8. Ghari, B., Tourani, A., Shahbahrani, A., Gaydadjiev, G.: Pedestrian detection in low-light conditions: a comprehensive survey. *Image Vis. Comput.*, 105106 (2024)
9. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
10. Han, C., Zhao, Q., Zhang, S., Chen, Y., Zhang, Z., Yuan, J.: Yolov2: better, faster, stronger for panoptic driving perception. arXiv preprint [arXiv:2208.11434](https://arxiv.org/abs/2208.11434) (2022)
11. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(9), 1904–1916 (2015). <https://doi.org/10.1109/TPAMI.2015.2389824>
12. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. *IEEE Intell. Syst. Appl.* **13**(4), 18–28 (1998)
13. Jiang, Q., Dai, J., Rui, T., Shao, F., Wang, J., Lu, G.: Attention-based cross-modality feature complementation for multispectral pedestrian detection. *IEEE Access* **10**, 53797–53809 (2022). <https://doi.org/10.1109/ACCESS.2022.3175303>
14. Jocher, G., Chaurasia, A., Qiu, J.: Ultralytics YOLO (Jan 2023). <https://github.com/ultralytics/ultralytics>
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. *Adv. Neural Inform. Process. Syst.* **25** (2012)
16. Li, C., et al.: Yolov6: a single-stage object detection framework for industrial applications. arXiv preprint [arXiv:2209.02976](https://arxiv.org/abs/2209.02976) (2022)
17. Lienhart, R., Maydt, J.: An extended set of haar-like features for rapid object detection. In: Proceedings, International Conference On Image Processing, vol. 1, pp. I–I. IEEE (2002)
18. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8759–8768 (2018)
19. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision* **60**(2), 91–110 (2004)

21. Ma, N., Zhang, X., Zheng, H.-T., Sun, J.: ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) *Computer Vision – ECCV 2018*. LNCS, vol. 11218, pp. 122–138. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_8
22. Mahaur, B., Singh, N., Mishra, K.: Road object detection: a comparative study of deep learning-based algorithms. *Multimedia Tools Appl.* **81**(10), 14247–14282 (2022)
23. Murthy, C.B., Hashmi, M.F., Keskar, A.G.: Efficientlitedet: a real-time pedestrian and vehicle detection algorithm. *Mach. Vis. Appl.* **33**(3), 47 (2022)
24. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 971–987 (2002)
25. Padilla, R., Netto, S.L., da Silva, E.A.B.: A survey on performance metrics for object-detection algorithms. In: 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 237–242 (2020). <https://doi.org/10.1109/IWSSIP48289.2020.9145130>
26. Qian, S., Ning, C., Hu, Y.: Mobilenetv3 for image classification. In: 2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), pp. 490–497. IEEE (2021)
27. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788 (2016)
28. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. *CoRR abs/arXiv:1612.08242* (2016)
29. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. *arXiv preprint arXiv:1804.02767* (2018)
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. *Adv. Neural Inform. Process. Syst.* **28** (2015)
31. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014)
32. Szegedy, C., et al.: Going deeper with convolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9 (2015)
33. Tan, M., Le, Q.: Efficientnetv2: smaller models and faster training. In: *International Conference on Machine Learning*, pp. 10096–10106. PMLR (2021)
34. Triki, N., Karray, M., Ksantini, M.: A comprehensive survey and analysis of traffic sign recognition systems with hardware implementation. *IEEE Access* (2024)
35. Verstraete, T., Muhammad, N.: Pedestrian collision avoidance in autonomous vehicles: a review. *Computers* **13**(3), 78 (2024)
36. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: Yolov7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696* (2022)
37. Wang, C.Y., Liao, H.Y.M., Wu, Y.H., Chen, P.Y., Hsieh, J.W., Yeh, I.H.: Cspnet: a new backbone that can enhance learning capability of cnn. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 390–391 (2020)
38. Wang, C.Y., Liao, H.Y.M., Yeh, I.H.: Designing network design strategies through gradient path analysis. *arXiv preprint arXiv:2211.04800* (2022)
39. Wang, C.Y., Yeh, I.H., Liao, H.Y.M.: Yolov9: learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616* (2024)

40. Wang, H., Liu, C., Cai, Y., Chen, L., Li, Y.: Yolov8-qsd: an improved small object detection algorithm for autonomous vehicles based on yolov8. *IEEE Trans. Instrument. Measur.* (2024)
41. Wang, J., Wu, Q.J., Zhang, N.: You only look at once for real-time and generic multi-task. *IEEE Trans. Vehic. Technol.* (2024)
42. Wu, D., et al.: Yolop: you only look once for panoptic driving perception. *Mach. Intell. Res.* **19**(6), 550–562 (2022)
43. Yu, F., et al.: Bdd100k: A diverse driving video database with scalable annotation tooling, vol. 2(5), p. 6. arXiv preprint [arXiv:1805.04687](https://arxiv.org/abs/1805.04687) (2018)
44. Zaidi, M., Daud, H., Shafique, M., Jamal, H.A.: Lane detection in autonomous driving: a comprehensive survey of methods and performance. In: 2024 IEEE 1st Karachi Section Humanitarian Technology Conference (KHI-HTC), pp. 1–6. IEEE



Machine Learning-Based System for Automated Crop Damage Detection and Classification in Insurance Claims

Abdou Karim Kandji¹✉, Kpêatchéhoué Merveille Santi Zinsou¹,
Thomas Henehan², and Jonathan Mwaura²

¹ University of Gaston Berger (UGB), BP: 234 Saint-Louis, Senegal
{kandji.abdou-karim1,zinsou.kpetchehoue-merveille-santi}@ugb.edu.sn

² Roux Institute at Northeastern University, Portland, ME 04101, USA
{Henehan.t,j.mwaura}@northeastern.edu

Abstract. Crop insurance is a critical tool for safeguarding farmers from financial losses caused by unforeseen events such as extreme weather, insect and pest infestations, and diseases. However, current claims processes including manual documentation and submission of damage evidence, are inefficient and slow. To address these challenges, this paper proposes an automated system that leverages machine learning techniques to classify and assess crop damage using images taken by farmers. This approach, known as picture-based insurance, enables rapid and accurate loss estimation based on image data submitted through a smartphone app. The reported work evaluated several machine learning models, including state-of-the-art architectures like Vision Transformers, ConvNeXt, and Swin Transformers, to classify different types of crop damage. In order to evaluate the model's performance, standard metrics to assess accuracy, precision, recall and f1-score were used. The experimental results demonstrate the effectiveness of the proposed approach in automating crop damage detection and classification, offering significant potential to improve the efficiency of crop insurance claims processing.

Keywords: insurance claims processing · machine learning · deep learning · automated damage classification

1 Introduction

Farmers often face a variety of challenges. These include adverse weather conditions (such as hail, storms, floods, and droughts etc.), insect and pests infestations, crop diseases, and fires. These events can severely damage crops, resulting in major economic losses. To safeguard farmers and agricultural producers from potential crop losses, it is crucial for them to obtain crop insurance. This type of

Supported by the United States Department of State (DoS): Embassy in Dakar, under the University Partnership Initiative, Grant No. SSG20023GR0032.

insurance provides essential protection against these agricultural disasters [13]. Currently, when farmers experience crop damage, they must document the damage and submit photographic evidence to their insurance companies for claims processing. This manual process takes a lot of time and labor-intensive, as it requires insurance adjusters to review and verify each image individually. The delays in processing claims can exacerbate the financial strain on farmers and slow down the delivery of necessary support and compensation.

This paper aims to develop an automated system for accurately identifying and classifying the specific types of damage in agricultural crops. This process called picture-based insurance [22] follows a succession of steps: firstly, photo capture by the farmer, secondly, each picture is automatically classified by a machine learning system to estimate losses, and finally decision making from insurance firms. The reported work utilises deep learning techniques. These models are known to require high performance computing and also are data intensive. In order to mitigate these issues, transfer learning approaches [37] are utilised.

The remainder of this article is organized as follows: related work on crop loss detection strategies is discussed in Sect. 2. This work also presents the dataset as well as preprocessing strategies in Sect. 3.1. In Sect. 3.2, the reported work presents the most advanced deep learning models used in the experiments. Section 3.4 discusses the machine learning training methods, as well as the experiments that were conducted. Further, Sect. 3 reports the results and discussions of conducted experiments. Finally, Sect. 5 draws the conclusion and offers insight for future work.

2 Related Work

Numerous studies across various global regions have investigated crop damage identification and classification using machine learning techniques. A notable distinction between these investigations is the type of data utilised, i.e. whether it is image or sensor data. This differentiation is crucial as the type of dataset directly influences model selection and performance, underscoring the need for tailored approaches for each category. In this section, a brief but detailed literature review is conducted specifically targeting work using image dataset and that utilising sensor data. Further, a subsection is added to discuss how the reported work draws from existing literature.

2.1 Crop Damage Classification Using Image-Based Datasets

Research using image datasets has focussed on images of different parts of the crop (such as leaves, fruit, stems etc.) and then creating classification models that learns whether a crop is damaged or not. For instance, Andrew J. et al. [19] utilized pre-trained CNNs to efficiently classify plant diseases. In their work, they fine-tuned models like DenseNet-121 [18], ResNet50 [16], VGG-16 [34], and Inception-v4 [36] applied to the PlantVillage dataset [11], which encompasses more than 54,000 images of plant diseases across 38 categories. Evaluation metrics including accuracy, sensitivity, specificity, and F1-score highlighted

DenseNet-121 as the most effective. Notably, DenseNet-121 achieved an accuracy score of 99.81%, surpassing other advanced methods.

Similarly, Divyanshu Tirkey et al. [38] released a real-time system for identifying, detecting, and classifying crop diseases. Their work focused on insect identification in soybean crops using neural network models. Their evaluation of transfer-learning models: YoloV5 yielded accuracy of 98.75%, followed by InceptionV3 with a score of 97%, and CNN with the same accuracy score (97%). YoloV5's rapid performance, operating at 53 fps, was noted for its suitability for quick detection. The authors collected a diverse crop insect images dataset in the aim of reducing the workload for producers and enhancing result effectiveness.

In another work, Mahmudul Hasan et al. [15] proposed a pipeline for crop and weed recognition that leverages generative adversarial networks (GANs) to enhance images, and segment them into overlapping patches. By selecting informative patches based on specific criteria and employing a weighted voting technique for classification, their method demonstrated significant improvements across ten deep learning models tested on four datasets: DeepWeeds [31], Cotton weed [5], Corn weed [20] and Cotton tomato weed [8]. DenseNet201 achieved top F1 scores of 98.49%, 99.83%, and 100%, while InceptionResNetV2 reached 98.96% on the Cotton weed dataset. This pipeline effectively addresses classification challenges and holds practical implications for farming.

2.2 Crop Damage Classification Using Sensor-Based Datasets

Studies involving sensor-based data investigate damage in a wider level across a farm or a geographical region. The data usually is collected using remote-sensing techniques over a period of time. For instance, Sosa L. et al. [35] applied machine learning algorithms to detect hailstorm damage in plant production and assess soil viability using time-series remote sensing data. The proposed tool aims to improve the accuracy and objectivity of damage assessments, benefiting all stakeholders in insurance contracts, including insurance companies, re-insurers, and policyholders. By relying on remote sensing data, the method reduces the need to extrapolate from limited on-site observations typically performed by insurance claims adjusters.

Similarly, Mann M. et al. [29] employed remote sensing and agricultural survey data to predict severe drought occurrences in Kenya. Fulkar B. et al. [10] used a CNN model with real-time field photography to classify agricultural parcels into those experiencing crop loss versus those unaffected. TKundu, N. et al. [23] proposed a severity scale for estimating maize crop loss. However, these approaches are often limited in scope, focusing narrowly on specific damage types like droughts or hailstorms.

2.3 Summary of Contributions and Gaps

The studies focusing on image datasets are more focused on individual crops in a farm. Images are collected and classified to determine damage. In the reviewed work, modeling is carried out using CNNs, which excel at capturing the intricate

details within visual data, making them highly suitable for classification tasks. Similarly, the work reported in this article uses CNNs as the baseline model and compares its efficiency with other deep learning variants.

Sensor-based datasets are employed for various machine learning tasks and mostly involve geographical regions or farmland. When the data is collected, there are numerous schemes to extract measurements for readiness in ML modeling. One technique is to rely on time-series analyses, as the data is collected over a period of time. Another approach, is to use the remote sensed imagery directly with deep learning techniques.

The work reported in this paper aims to classify crop damage over a wide geographic area. As such, image datasets as reviewed in Sect. 2.1 are not sufficient for this type of classification. Further, remote-sensed data which would be ideal is often out of reach for farmers and practitioners. As such, the reported work seeks to utilise images of farmland (geographical areas) collected using cellphone cameras. Note that, these are not images of individual crops but of an area/farmland and therefore somewhat mirrors remote-sensed data. The data can then be used to address a wider range of crop damage scenarios and enhance accessibility for damage identification. At the time of writing, the authors are not familiar with any ongoing or past work that has utilised this type of dataset in a previously documented work¹.

3 Materials and Methods

This section presents and details the dataset utilized in the reported work. In addition, the baseline model and advanced models used during the reported experimentation phase, are discussed.

3.1 Dataset

The work presented in this paper utilizes the Eyes on the Ground Image Data dataset [40], which was collaboratively developed by ACRE Africa, the International Food Policy Research Institute (IFPRI), and the Lacuna Fund. The dataset aims to support the development of comprehensive machine learning (ML) models for smallholder farmers' fields, building upon prior research from the Picture-Based Insurance framework [4].

This dataset is distinctive as it contains location-tagged crop images paired with detailed labels on input usage, agricultural management practices, growth stages, crop damage, and harvest yields. Data were collected across eight counties in Kenya, leveraging the GPS functionality of smallholder farmers' cellphones for geolocation. To ensure privacy, the exact GPS coordinates were excluded, and only the bounding boxes of villages and village names, sourced from the

¹ The authors note that a challenge on crop damage identification using this dataset has been reported on Zindi (<https://zindi.africa/competitions/cgiar-crop-damage-classification-challenge>). However, there is no record of published work.

GADM36 dataset², were provided. Labels classify images based on crop growth stages, types, and severity of damage, including disruptions from drought, weeds, and pests. Expert agronomists or trained professionals manually labeled the cell-phone images to ensure quality and consistency. In cases where manual labels were unavailable, automatic labels were generated using an in-house ML algorithm.

The dataset is split into a training set, with labels provided in a “train.csv” file, and a test set, which lacks labels. The train set contains approximately 26,000 images, categorized into damage types: drought (DR), good growth (G), nutrient deficiency (ND), weed (WD), and other damages (including diseases, pests, and wind). Table 1 defines these categories.

Table 1. Damage types and definitions.

Damage	Definition
DR	Drought
G	Good (growth)
ND	Nutrient Deficient
WD	Weed
other	Disease, Pest, Wind

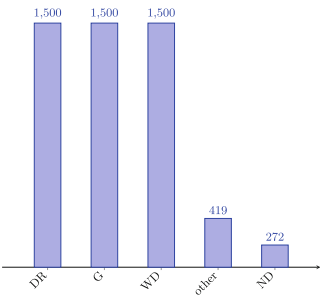


Fig. 1. Number of samples by categories.

Due to resource and computational constraints, this study used a subset of the dataset. Observations were categorized and distributed as shown in Fig. 1, with the *RandomUnderSampler* class from Python’s *imblearn* library [24] employed to balance the dataset by downsampling majority classes. The classes remain unbalanced, with minority categories like ND and other being under-represented. The subset was divided into training set (80%) and test set (20%), resulting in 4,152 pictures in the training data and 1,039 images in the test data.

² <https://gadm.org/index.html>.

Before inputting pictures into the model, several preprocessing strategies were applied to the training set. These involve data transformations such as: *resizing* the images to a specified height and width, *normalizing* tensor images with mean and standard deviation for each channel. Furthermore data augmentations such as *rotating* the images by a specified angle and randomly flipping them horizontally with a set probability. Additionally, the *brightness* level and *contrast* were randomly adjusted to enhance data variability.

3.2 Deep ML Techniques

Baseline Model: Resnet50 [16] based on CNN [9] and illustrated in Fig. 2 was considered as a baseline model in this work. Convolutional Neural Networks (CNNs) are particularly relevant from other types of neural networks due to their better performance in image-related tasks. A CNN is typically composed of 3 primary layers: the Convolutional layer, the Pooling layer, and the Fully-connected (FC) layer. The initial layers of a CNN are designed to detect simple features of an image, such as colors and edges, while the deeper layers are capable of identifying more complex structures and larger patterns. This hierarchical feature extraction process enables the network to accurately recognize objects within an image.

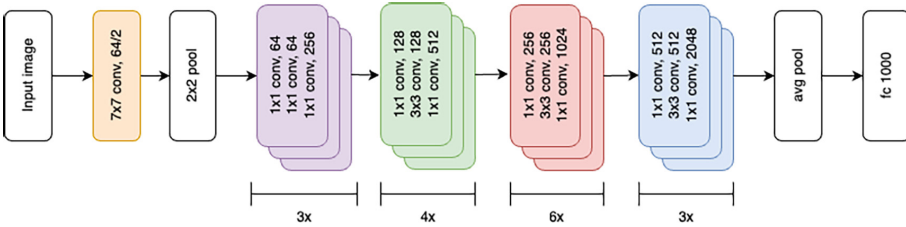


Fig. 2. ResNet50 architecture

The convolutional layer, a key component of the network, consists of three primary elements: the input data, the filter, and the feature map. The input data, which represents the pixel matrix of the image, is convolved with a two-dimensional filter. The resulting operation is the feature map, which highlights relevant features detected by the filter. Following this, the pooling layer reduces the dimensionality of the feature maps, thereby minimizing computational complexity and reducing the risk of overfitting. Finally, the FC layer processes extracted features to perform classification, assigning the image to a specific category based on the features learned from the previous layers. This combination of convolution, pooling, and classification enables CNNs to excel in object recognition and image classification tasks.

The particularity of ResNet50 is to avoid the problem of vanishing gradient [2, 14] which deep CNNs such as VGG16 presented by Simonyan et al. [34] suffer

from. Vanishing gradients are caused by the non-updating of gradients due to the depth of neural networks. This results in a slowdown in the learning process or an inability to learn. To solve this problem, ResNet applies the notion of skip connection in order to preserve the information of the first convolution layers in the entire network. This technique is applied in a block named Bottleneck Residual Block which consists of ReLU (Rectified Linear Unit) activation function [14], convolution layers, batch normalization layers and a Skip Connection [16].

Vision Transformer (ViT): The ViT model [7] illustrated in Fig. 3 and based on Transformer [39] receives as input an image split into patches of fixed size. Each of the segments is passed through a linear embedding, then into a position embeddings layer. The resulting vector sequence is then introduced into a Transformer encoder. Note that a token used for image classification is added to the Transformer input sequence. A Multi-Layer Perceptron layer is then added on the top of the Transformer encoder for image classification.

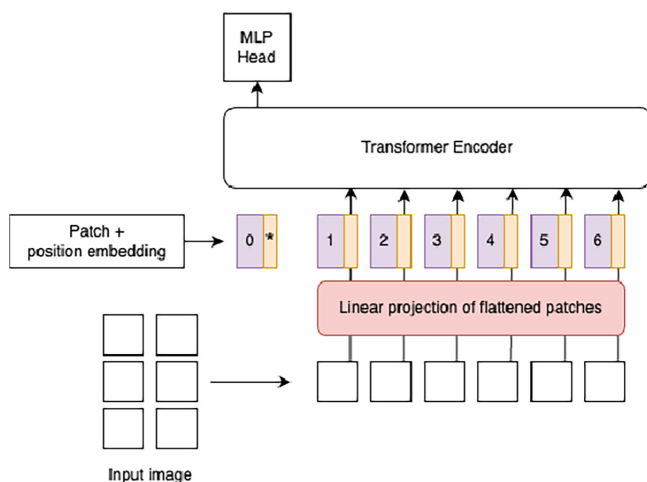


Fig. 3. ViT architecture

Swin Transformer: This model [26] is an evolved version of ViT that introduces the notion of shifted windows where the model learns to capture relationships between different parts of the image. This is carried out by varying the size of the windows after each successive Swin Transformer blocks. The ViT model uses self-attention mechanism to compute the relationships between a token and every other token. On the other hand, Swin Transformer uses the self-attention layer to learn various regions of the image by varying the size of the windows

between two blocks. This strategy reduces the computational complexity when extracting features from the image. The architecture as illustrated in Fig. 4 is structured in 4 stages: Stage 1 comprises of a linear embedding and Swin Transformer block while second, third, and fourth stages comprises of patch merging stage followed by Swin Transformer block.

In the first stage, inputted image is partitioned into patches of same size. Each partition is of size $(H/4 \times W/4)$, where H and W represent respectively the height and the width of each patch. Subsequently, every patch is transformed using a linear layer projected on a dimension (denoted C). The resulting feature map with dimension size $H/4 \times W/4 \times C$ is then passed through a Swin Transformer block which is made up of twin Transformer encoder blocks (shown by Fig. 5).

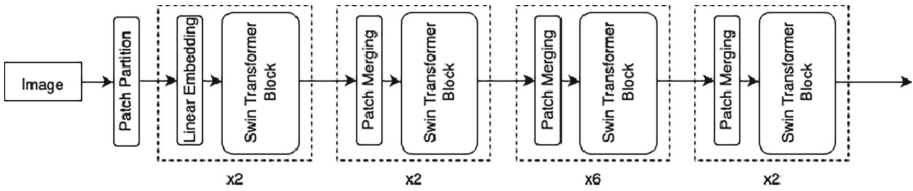


Fig. 4. Swin Transformer architecture

What differentiates Swin Transformer block from a classic Transformer encoder is the replacement of the Multi-head Attention module by the *Windowed multi-headed self-attention* (W-MSA) see Fig. 5 (left block) and Eq. 1. W-MSA consists of dividing the input image into smaller, separate, non-overlapping regions. The self-attention mechanism is then applied separately inside each window, meaning that the model only looks at relationships between patches inside each window. In the second block (see Fig. 5 (right block) and Eq. 2), the *Self Attention Across “Shifted” Windows* (SW-MSA), allows the model to look at relationships across the boundaries of the previous windows, capturing connections between patches that were not in the same window before. The resulting vector Z of consecutive Swin Transformer blocks is computed with Eq. 3.

The following stage combines the result of the primary stage with the neighboring window, then down-samples the resolution by $2x$ and increases the feature map depth by 2 through a linear layer. This process is called patch merging. The generated feature map is subsequently fed through a Swin Transformer block. Stages 3 and 4 repeat the operations of stage 2.

$$\text{W-MSA} = 4hwC^2 + 2(hw)^2C \quad (1)$$

$$\text{SW-MSA} = 4hwC^2 + 2M^2hwC \quad (2)$$

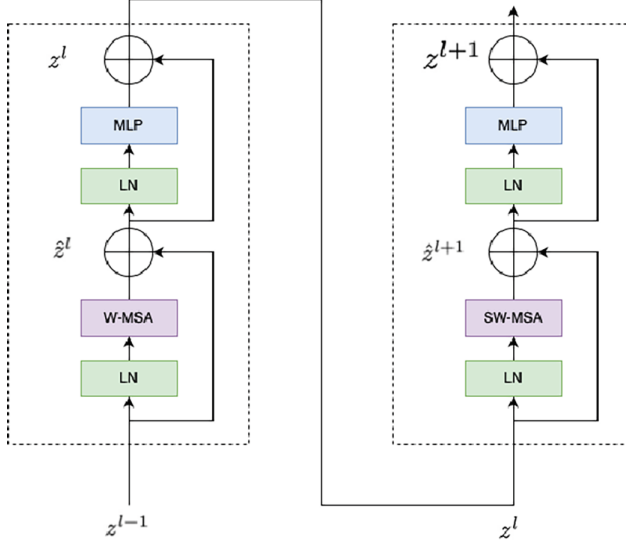


Fig. 5. Two successive Swin Transformer Block

$$\hat{Z}^l = \text{W-MSA}(\text{LN}(Z^{l-1})) + Z^{l-1}, \quad (3)$$

$$Z^l = \text{MLP}(\text{LN}(\hat{Z}^l)) + \hat{Z}^l, \quad (4)$$

$$\hat{Z}^{l+1} = \text{SW-MSA}(\text{LN}(Z^l)) + Z^l, \quad (5)$$

$$Z^{l+1} = \text{MLP}(\text{LN}(\hat{Z}^{l+1})) + \hat{Z}^{l+1} \quad (6)$$

where h and w are the width and height of the input image. C represents the number of channels in the hidden layers of the first stage. MLP is a Multi-Layer Perceptron and LN is a Layer Normalization.

ConvNeXt: The ConvNeXt model [27] primarily consists of CNNs followed by FC layers. A distinctive feature of ConvNeXt is its implementation of a "global average pooling" layer [25]. This layer compresses the features produced by the convolutional layers into a single vector by averaging their spatial dimensions. The resulting vector is then updated into the FC layers for classification or regression tasks. To avoid overfitting, ConvNeXt uses Layer Normalization (LN) and non-linear activation layers such as Gaussian Error Linear Units (GELU) [17]. The general architecture of ConvNeXt is a succession of 4 ConvNeXt blocks illustrated in Fig. 6. Every block is separated by a 2×2 conv layer. The top of the network has an average pooling layer and an FC layer.

3.3 Performance Measures

This section presents each metric used to evaluate all the models employed in the reported experimentation.

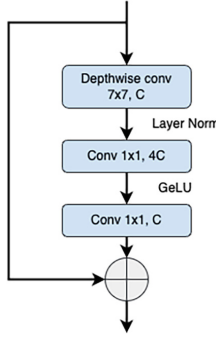


Fig. 6. ConvNeXt Block

Accuracy: This metric is used to assess the performance of a classifier. It measures how often the model accurately predicts the correct label. The formula for accuracy is:

$$accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(y = \hat{y}) \quad (7)$$

where y is the correct class, $\hat{y}$ represents the predicted class and n represents the number of samples.

Precision: This metric is used to assess the classifier's ability to identify all positive samples among those predicted as positive. This metric helps to answer the question: Of all the instances classified as positive, how many are truly positive?

$$precision = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \quad (8)$$

Here, *True Positives* are cases where the model accurately identifies the positive class, while *False Positives* are cases where the model incorrectly predicts a positive class.

Recall (Sensitivity) : This metric is used to evaluate the ability of the classifier to predict all the positive samples. Sensitivity helps to answer: Of all the actual positive samples, how many did the model truly identify?

$$recall = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \quad (9)$$

where *False negatives* refer to instances where the model fails to predict a specific class correctly.

F1-Score: This metric is the harmonic mean of precision and recall, with its best possible value being 1 and the worst being 0. The F1-score is used to provide a balanced assessment of a model's performance when there is an uneven

distribution between the classes and also when precision and recall need to be balanced.

$$\text{F1-score} = 2 * \frac{\text{True positives}}{2 * \text{True positives} + \text{False positives} + \text{False negatives}} \quad (10)$$

3.4 Model Training

Pytorch library [32] was used to fine-tune the Resnet50 model. All experiments were run on Google Colab notebooks [1]. T4 GPU was used for Resnet50 and L4 GPU for ViT, SwinTransformer and ConvNeXt finetuning. For ResNet50, input images were resized to 255×255 , and several transformations were applied to the training set, including RandomResizedCrop, RandomHorizontalFlip, ColorJitter, and Normalization. For the test set, only CenterCrop and Normalization transformations were applied.

For ViT, the training set underwent RandomResizedCrop and Normalization transformations. The SwinTransformer model used RandomResizedCrop, RandomHorizontalFlip, and Normalization for training, with CenterCrop and Normalization for testing. Finally, the ConvNeXt model applied RandomRotate90, HorizontalFlip, RandomBrightnessContrast, and Normalization to the training set, while the test set was limited to Resizing and Normalization.

ResNet50 was pre-trained on ImageNet dataset and fine-tuned on the utilised dataset. All parameters were frozen except the last fully connected layers by adding a Linear layer, a ReLU activation function, a Dropout, and a LogSoftmax for classification. The negative log likelihood loss function (NLLLoss) was used during fine-tuning process. Adam [21] was used as a technique for parameter optimization.

ViT was pre-trained on ImageNet dataset, and fine-tuned using the Transformers library [41] on top of pre-trained model provided by Google (called vit-base-patch16-224-in21k) available on Hugging Face. During training, the AdamW optimizer [28] was used to update the model's weights. Similarly, Swin Transformer and ConvNeXt models were also pre-trained on the same ImageNet dataset, and fine-tuned using the same process as for ViT. The hyperparameters used for fine-tuning each model are reported in Table 2.

4 Result and Discussion

Accuracy, Precision, Recall, and F1-score, as described in Sect. 3.3, were used to evaluate and compare each model's performance. The results for all models across different damage categories are summarized in Table 3.

Table 4 shows the weighted average results for each model in terms of damage classification. The weighted average of recall, precision, and F1-score is computed for multi-class classification problems to account for the different number of instances in each class. The formula for the weighted average of a metric (recall, precision, or F1-score) is:

Table 2. Hyperparameters for all models.

model	learning rate	# of epochs	batch size	optimizer
ResNet50	3e-3	10	64	Adam
ViT	5e-5	10	16	AdamW
SwinTransformer	5e-5	10	32	AdamW
ConvNeXt	5e-5	10	32	AdamW

Table 3. Classification result.

Model	Label	Precision	Recall	F1-score
ResNet50	DR	0.58	0.72	0.65
	G	0.55	0.43	0.48
	ND	0.00	0.00	0.00
	WD	0.52	0.69	0.59
	other	0.61	0.24	0.34
ViT	DR	0.69	0.79	0.74
	G	0.67	0.65	0.66
	ND	0.56	0.16	0.25
	WD	0.69	0.73	0.71
	other	0.58	0.51	0.54
SwinTransformer	DR	0.71	0.79	0.75
	G	0.64	0.65	0.65
	ND	0.42	0.09	0.15
	WD	0.65	0.72	0.68
	other	0.66	0.44	0.53
ConvNeXt	DR	0.66	0.80	0.72
	G	0.52	0.71	0.60
	ND	1.00	0.02	0.04
	WD	0.69	0.49	0.57
	other	0.45	0.30	0.36

$$\text{Weighted Average} = \frac{1}{N} \sum_{i=1}^n w_i \cdot M_i$$

where n represents the number of classes, M_i is the metric (precision, recall, or F1-score) for class i , w_i is the weight for class i , calculated as the number of instances in class i , N represents the total number of instances across all classes.

The results in Table 3 demonstrate that the ViT model outperformed all other models in terms of F1-score across all damage types, except for Drought

Table 4. Results.

Model	Accuracy	Precision	Recall	F1-score
ResNet50	0.55	0.53	0.55	0.52
ViT	0.67	0.67	0.67	0.66
SwinTransformer	0.66	0.66	0.66	0.65
ConvNeXt	0.63	0.64	0.63	0.61

(DR). Table 4 further highlights the superiority of ViT across various performance metrics, particularly in general damage classification. ViT achieved the highest F1-score of 66%, followed by Swin Transformer and ConvNeXt models with scores of 65% and 61%, respectively, significantly outperforming the ResNet baseline. However, the model showed weaker performance on the Nutrient Deficient (ND) category, which can be attributed to the smaller sample size for this class compared to the others.

To compare advanced models with the baseline, statistical significance tests were conducted on the test split. Since McNemar’s statistical test [30] is limited to binary classes, the non-parametric McNemar-Bowker test [3] was used to evaluate whether two models differ significantly in their categorization of the same data. This test is particularly suited for computationally expensive models, such as large deep learning architectures. The null hypothesis assumes no significant difference in the predictions of the two classifiers on the test set. A p-value below 0.05 determines that the null hypothesis can be rejected, meaning one model significantly outperforms the other. Table 5 shows a significant difference between ViT and the ResNet50 baseline, as well as between ViT and ConvNeXt, with p-values lower than 0.05 based on the McNemar-Bowker test.

Table 5. McNemar-Bowker Test for each classifier compared to ViT.

Model	chi-square	p-value
ResNet50	79.272	6.9719e-13
SwinTransformer	17.829	0.0579
ConvNeXt	29.948	0.0008

In the reported results, The p-value was below the significance level (0.05), resulting in the rejection of the null hypothesis. This indicates that ViT, ResNet50, and ConvNeXt make significantly different predictions on the same dataset, with ViT achieving a higher F1-score compared to ResNet50 and ConvNeXt. This confirms, with 95% confidence, that ViT outperforms ResNet50 and ConvNeXt. However, the null hypothesis could not be rejected for ViT and Swin Transformer, as the p-value was slightly greater than 0.05. Therefore, it cannot be concluded that ViT is significantly better than Swin Transformer.

Model Interpretability. To enhance the interpretability of the reported results, the *pytorch-grad-cam* tool [12] was used, incorporating the Deep Feature Factorization (DFF) technique described by [6]. DFF identifies similar semantic concepts within an image or across multiple images by applying non-negative matrix factorization (NMF) to the activations of a deep CNN layer. This approach reveals hierarchical clusters in the feature space that correspond to semantic concepts, which can then be visualized as heatmaps to show what the network perceives as similar.

Grad-CAM visualizations in Fig. 7 showcase these concepts, using different colors to represent distinct classes. For example, Fig. 7a highlights good growth is represented in green and weed is in blue, while Fig. 7b identifies weed-affected areas in green. In Fig. 7c, drought-related damage is shown in red. These visualizations offer valuable insights into the model’s classification process and its perception of features within the dataset.

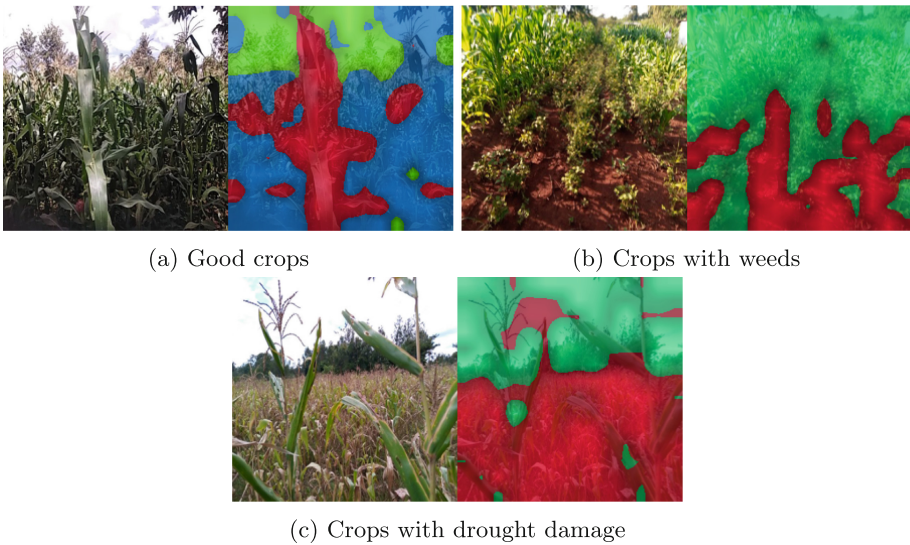


Fig. 7. DFF heat maps for G, WD and DR images

Ablation Study. To evaluate the impact of data augmentation, the *RandomResizedCrop* feature was replaced with the *Resize* feature in one simulation. In another, *normalization* was removed entirely. Each model was trained over 10 epochs on the same test set, and the resulting F1-scores are presented in Table 6. The first model, without *RandomResizedCrop*, achieved an F1-score of 62%, while the second, without *normalization*, scored 63%. These versions showed respective gaps of 4% and 3% compared to the baseline. A third model was trained using the Adafactor optimizer [33] instead of AdamW, resulting in significantly poorer performance with an F1-score of just 18%.

Table 6. F1-score of ablation studies.

Model	f1-score
w/o RandomResizedCrop	0.62
w/o Normalize	0.63
w Adafactor	0.18
Baseline (ViT)	0.66

5 Conclusion

The reported work presents an automated system for classifying crop damage using photos captured by smartphones. The aim is to streamline and enhance the accuracy of crop insurance claims processing. The reported approach leverages a diverse dataset of georeferenced images from smallholder farms across eight counties in Kenya, encompassing various crop damage types and severities.

The reported work evaluated several machine learning models, including ResNet50, Vision Transformer, Swin Transformer, and ConvNeXt, to determine their effectiveness in classifying crop damage. The results demonstrated that advanced models, particularly the Vision Transformer (ViT), outperformed the ResNet50 baseline, achieving the highest F1-score of 66%. This performance highlights the potential of using state-of-the-art models for image-based crop damage classification, providing a significant improvement over traditional methods.

The study’s findings underscore the effectiveness of leveraging advanced machine learning models and smartphone technology to automate and expedite crop insurance claims. By reducing the reliance on manual processing and offering real-time damage assessment, the proposed system has the potential to alleviate financial strain on farmers and enhance the efficiency of insurance claims processing.

Future work will aim to expand the dataset to incorporate more nutrient deficient and other damage scenarios such as disease, pest or wind with data augmentation. This will help ensure the system’s robustness and adaptability across various agricultural settings and contribute to more sustainable and resilient agricultural practices.

References

1. Welcome to Colaboratory - Colab. <https://colab.research.google.com/>
2. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **5**(2), 157–166 (1994). <https://doi.org/10.1109/72.279181>
3. Bowker, A.H.: A test for symmetry in contingency tables. *J. Am. Stat. Assoc.* **43**(244), 572–574 (1948)

4. Ceballos, F., Kramer, B., Robles, M.: The feasibility of picture-based insurance (pbi): smartphone pictures for affordable crop insurance. *Developm. Eng.* **4**, 100042 (2019)
5. Chen, D., Lu, Y., Li, Z., Young, S.: Performance evaluation of deep transfer learning on multi-class identification of common weed species in cotton production systems. *Comput. Electr. Agricult.* **198**, 107091 (2022). <https://doi.org/10.1016/j.compag.2022.107091>, <https://www.sciencedirect.com/science/article/pii/S0168169922004082>
6. Collins, E., Achanta, R., Süsstrunk, S.: Deep feature factorization for concept discovery (2018). <https://arxiv.org/abs/1806.10206>
7. Dosovitskiy, A., et al.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (Jun 2021). <https://doi.org/10.48550/arXiv.2010.11929>
8. Espejo-Garcia, B., Mylonas, N., Athanasakos, L., Fountas, S., Vasilakoglou, I.: Towards weeds identification assistance through transfer learning. *Comput. Electron. Agric.* **171**, 105306 (2020)
9. Fukushima, K.: Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybernet.* **36**(4), 193–202 (1980)
10. Fulkar, B., Patil, P., Srivastav, G., Mahale, P.: Predicting agricultural crop damage caused by unexpected rainfall using deep learning, pp. 1–5 (2024). <https://doi.org/10.1109/ISCS61804.2024.10580999>
11. Geetharamani, G., Arun Pandian, J.: Identification of plant leaf diseases using a nine-layer deep convolutional neural network. *Comput. Electr. Eng.* **76**, 323–338 (2019). <https://doi.org/10.1016/j.compeleceng.2019.04.011>, <https://www.sciencedirect.com/science/article/pii/S0045790619300023>
12. Gildenblat, J., contributors: Pytorch library for cam methods (2021). <https://github.com/jacobgil/pytorch-grad-cam>
13. Glauber, J.W.: Crop insurance reconsidered. *Am. J. Agricult. Econ.* **86**(5), 1179–1195 (2004). <http://www.jstor.org/stable/3697927>
14. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: International Conference on Artificial Intelligence and Statistics (2010). <https://api.semanticscholar.org/CorpusID:5575601>
15. Hasan, A., Diepeveen, D., Laga, H., Jones, M., Sohel, F.: Image patch-based deep learning approach for crop and weed recognition. *Eco. Inform.* **78**, 102361 (2023)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition (Dec 2015). <https://arxiv.org/abs/1512.03385v1>
17. Hendrycks, D., Gimpel, K.: Gaussian Error Linear Units (GELUs) (Jun 2023). <https://doi.org/10.48550/arXiv.1606.08415>
18. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely Connected Convolutional Networks (Aug 2016). <https://arxiv.org/abs/1608.06993v5>
19. J., A., Eunice, J., Popescu, D.E., Chowdary, M.K., Hemanth, J.: Deep learning-based leaf disease detection in crops using images for agricultural applications. *Agronomy* **12**(10), 2395 (2022). <https://doi.org/10.3390/agronomy12102395>, <https://www.mdpi.com/2073-4395/12/10/2395>
20. Jiang, H., Zhang, C., Qiao, Y., Zhang, Z., Zhang, W., Song, C.: CNN feature based graph convolutional network for weed and crop recognition in smart farming. *Comput. Electr. Agricult.* **174**, 105450 (Jul 2020). <https://doi.org/10.1016/j.compag.2020.105450>, <https://www.sciencedirect.com/science/article/pii/S0168169919321349>
21. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (Jan 2017). <https://doi.org/10.48550/arXiv.1412.6980>

22. Kramer, B.: Climate insurance: Opportunities for improving agricultural risk management in Kenya (Dec 2023). <https://hdl.handle.net/10568/138995>
23. Kundu, N., et al.: Disease detection, severity prediction, and crop loss estimation in maizecrop using deep learning. *Artificial Intell. Agricult.* **6**, 276–291 (2022)
24. Lemaître, G., Nogueira, F., Aridas, C.K.: Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. *J. Mach. Learn. Res.* **18**(17), 1–5 (2017). <http://jmlr.org/papers/v18/16-365.html>
25. Lin, M., Chen, Q., Yan, S.: Network In Network (Mar 2014). <https://doi.org/10.48550/arXiv.1312.4400>
26. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows (Aug 2021). <https://doi.org/10.48550/arXiv.2103.14030>
27. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A ConvNet for the 2020s (Mar 2022). <https://doi.org/10.48550/arXiv.2201.03545>
28. Loshchilov, I., Hutter, F.: Decoupled Weight Decay Regularization (Jan 2019). <https://doi.org/10.48550/arXiv.1711.05101>
29. Mann, M.L., Warner, J.M., Malik, A.S.: Predicting high-magnitude, low-frequency crop losses using machine learning: an application to cereal crops in Ethiopia. *Clim. Change* **154**(1), 211–227 (2019)
30. McNemar, Q.: Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* **12**(2), 153–157 (1947)
31. Olsen, A., et al.: DeepWeeds: a multiclass weed species image dataset for deep learning. *Sci. Rep.* **9**(1), 2058 (2019)
32. Paszke, A., et al.: PyTorch: an Imperative Style, High-Performance Deep Learning Library (Dec 2019). <https://doi.org/10.48550/arXiv.1912.01703>
33. Shazeer, N., Stern, M.: Adafactor: adaptive learning rates with sublinear memory cost (2018). <https://arxiv.org/abs/1804.04235>
34. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition (Apr 2015). <https://doi.org/10.48550/arXiv.1409.1556>
35. Sosa, L., Justel, A., Molina, I.: Detection of crop hail damage with a machine learning algorithm using time series of remote sensing data. *Agronomy* **11**(10) (2021). <https://doi.org/10.3390/agronomy11102078>, <https://www.mdpi.com/2073-4395/11/10/2078>
36. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning (Aug 2016). <https://doi.org/10.48550/arXiv.1602.07261>
37. Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C.: A survey on deep transfer learning. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds.) *ICANN 2018. LNCS*, vol. 11141, pp. 270–279. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01424-7_27
38. Tirkey, D., Singh, K.K., Tripathi, S.: Performance analysis of AI-based solutions for crop disease identification, detection, and classification. *Smart Agricult. Technol.* **5**, 100238 (2023)
39. Vaswani, A., et al.: Attention is All you Need. In: *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc. (2017). https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
40. Waithaka, L., Kramer, B., Hufkens, K., Kivuva, B., Mansabdar, S.: Eyes on the Ground Image Data (2022). <https://doi.org/10.34911/rdnt.1bs2jw>
41. Wolf, T., et al.: HuggingFace’s Transformers: State-of-the-art Natural Language Processing (Jul 2020). <https://doi.org/10.48550/arXiv.1910.03771>

Author Index

A

Abba Ari, Ado Adamou II-144
Abisoye, Blessing Olatunde II-291
Aboagye, Manhyia Gideon II-275
Aboagye, Sylvester I-240
Agyemang, Justice Owusu I-152, II-275
Ahiaklo-Kuz, Raphael I-240
Akoguhi, N'guessan Patrice I-210
Aliyu, Ahmed II-194
Alsolame, Tarek I-193
Amaury, Cansa Kayembe II-177
Antogiovanni, Stephen I-193
Aritra, B. D. S. I-193
Assohoun, Egomli Stanislas I-210
Atakpla, Desmond Delali I-152

B

Ba, Cheikh II-105
Bagha, Amir Mohammadi II-164
Bagula, Antoine I-83, II-260
Boadu, Thomas I-152
Bokaba, Tebogo I-275, II-70, II-194
Bueno, Felipe I-252
Byenkyia, Nkalubo Lenard II-17, II-125

C

Chakrabortii, Chandranil I-193
Cheng, Ling I-228
Clinton, Addai II-275

D

Darko, Christabel Bemah I-152
Diakhame, Mouhamadou Lamine II-213
Diattara, Awa II-105
Dieng, Diarra Bousso II-213
Dogo, Eustace M. II-194
Dombeu, Jean Vincent Fonou II-36, II-87
Dtissibe, Francis Yongwa II-144
Du Plessis, Deon I-133

E

Elikplim, Henyo Chester II-275

G

Gueroui, Abdelhak Mourad II-144
Gwetu, Mandlenkosi V. I-97
Gwetu, Mandlenkosi Victor II-36
Gwetu, Mandlenkosi II-55, II-87, II-227

H

Halefose, Kholofelo H. I-133
Hassanalizadeh, Azin II-164
Henehan, Thomas II-369

I

Igboamalu, Frank I-228

J

Jonathan, Kabemba Ntumbwa II-177

K

Kandji, Abdou Karim II-369
Kodua-Basoa, Martin I-152
Kogeda, Okuthe P. I-34
Koua, Kpangni Alex Jérémie I-210, II-326
Kouame, Koffi Fernand I-210
Kouame, N'dri Pascal I-210
Kparbong, Ponjul Bali II-194
Kwenda, Clopas II-36
Kyangwa, Henriette I-83

L

Leke, Collins A. I-3
Lekola, Bafokeng II-347
Lekola, Serestina II-347
Lerato, Moila Ramahlapane I-177
Lumamba, Kapongo D. I-97

M

Maguraushe, Kudakwashe I-275
 Makun, Hussaini A. II-194
 Mathonsi, Topside E. I-133
 Mbale, Landry I-83, II-260
 Merlec, Mpyana Mwamba II-177
 Mhlongo, Siyabonga II-70
 Mienye, Ibomoie Domor II-3
 Mngomezulu, Mangaliso II-55
 Molefe, Mohale II-309
 Moodley, Larisha I-34
 Msimnag, Mpumelelo I-97
 Mthulisi, Velepini I-177
 Muchenje, Tonderai I-133
 Mwaura, Jonathan I-121, II-213, II-326,
 II-369

N

Naidoo, Threnesan I-97
 Ndayizigamiye, Patrick I-275
 Ndjiongue, Alain R. I-3
 Ndjiongue, Alain I-228
 Ndoundam, René II-144
 Ngandu, Bernard II-260
 Ngossaha, Justin Moskolai II-144
 Nixson, Okila II-17, II-125
 Niyonkuru, Enock I-193
 Nougaret, Joshua II-213
 Nyasulu, Chimango I-121

O

Obaido, George II-3
 Obed, Somuah II-275
 Ofusori, Lizzy II-70
 Olawuyi, Olushola I-15, I-54
 Ouahada, Khmaies I-228
 Owolawi, Pius I-166

P

Pouamoun, Ibrahim Aoudou II-144

R

Rawat, Danda B. II-164
 Rose, Nakibuule II-17, II-125
 Rotshidzwa, Phandavhudzi I-177
 Rundensteiner, Elke II-105

S

Saghir, Zohaib I-240
 Sané, Fatou Cire Emilie II-326
 Selorm, Awumee Gabriel I-152, II-275
 Seretloa, Semakaleng I-166
 Sibanda, Khulumani I-72
 Sibitenda, Harriet II-105
 Singh, Yuvika I-34
 Steyn, Adrie J. C. I-97
 Sun, Yanxia II-291
 Swart, Theo G. II-3

T

Tamuka, Nyashadzashe I-72
 Tanwar, Sudeep II-164
 Tchaleu, Boriane Y. I-3
 Tivir, Cecilia I-121
 Traore, Assitan II-105
 Traore, Issa II-164
 Tu, Chunling I-166

V

Vambe, Trevor II-227
 Viriri, Serestina II-227

W

Warren, Michael II-326
 Wells, Gordon I-97
 Woungang, Isaac II-164

Z

Zengeya, Tsitsi II-87
 Zenghui, Wang II-291
 Zinsou, Kpêtchéhoué Merveille Santi II-369