Lecture Notes in Electrical Engineering 1299

Nima Dokoohaki
Julia Laserre
Reza Shirvany *Editors*

Revolutionizing Fashion and Retail

Proceedings of the Fifth FashionXrecsys Workshop at the Recommender Systems Conference, Singapore, 18th–22nd September 2023

Springer

Lecture Notes in Electrical Engineering

Volume 1299

The book series *Lecture Notes in Electrical Engineering* (LNEE) publishes the latest developments in Electrical Engineering—quickly, informally and in high quality. While original research reported in proceedings and monographs has traditionally formed the core of LNEE, we also encourage authors to submit books devoted to supporting student education and professional training in the various fields and applications areas of electrical engineering. The series cover classical and emerging topics concerning:

- Communication Engineering, Information Theory and Networks
- Electronics Engineering and Microelectronics
- Signal, Image and Speech Processing
- Wireless and Mobile Communication
- Circuits and Systems
- Energy Systems, Power Electronics and Electrical Machines
- Electro-optical Engineering
- Instrumentation Engineering
- Avionics Engineering
- Control Systems
- Internet-of-Things and Cybersecurity
- Biomedical Devices, MEMS and NEMS

For general information about this book series, comments or suggestions, please contact leontina.dicecco@springer.com.

To submit a proposal or request further information, please contact the Publishing Editor in your country:

China

Jasmine Dou, Editor (jasmine.dou@springer.com)

India, Japan, Rest of Asia

Swati Meherishi, Editorial Director (Swati.Meherishi@springer.com)

Southeast Asia, Australia, New Zealand

Ramesh Nath Premnath, Editor (ramesh.premnath@springernature.com)

USA, Canada

Michael Luby, Senior Editor (michael.luby@springer.com)

All other Countries

Leontina Di Cecco, Senior Editor (leontina.dicecco@springer.com)

**** This series is indexed by EI Compendex and Scopus databases. ****

Nima Dokoohaki · Julia Laserre · Reza Shirvany
Editors

Revolutionizing Fashion and Retail

Proceedings of the Fifth FashionXrecsys Workshop at the Recommender Systems Conference, Singapore, 18th–22nd September 2023

Springer

Editors
Nima Dokoohaki
Accenture SAS (France)
Biot, France

Julia Laserre
Zalando SE (Germany)
Berlin, Germany

Reza Shirvany
Zalando SE (Germany)
Berlin, Germany

ISSN 1876-1100 ISSN 1876-1119 (electronic)
Lecture Notes in Electrical Engineering
ISBN 978-3-031-76877-4 ISBN 978-3-031-76878-1 (eBook)
https://doi.org/10.1007/978-3-031-76878-1

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

Acknowledgments

This effort has been possible with the help of the organization team of the workshop on Recommender Systems in Fashion, namely, Julia Lasserre, Nima Dokoohaki, and Reza Shirvany. Editors would like to gratefully acknowledge the support from past editions' organization members Shatha Jaradat and Humberto Corona, and that of the technical committee of this workshop's edition, namely, Alexey Kurennoy, Amrollah Seifoddini, Anoop Katti, Eder Martins, Hamedeh Jafari, Hareesh Bahuleyan, Humberto Corona, Jelle Stienstra, Jonas Pollok, Leonidas Lefakis, Luís Baía, Marjan Celikik, Mustafa Khandwawala, Nikolay Jetchev, Pedro Nogueira, Sahan Ayvaz, Sebastian Heinz, Sonia Aurelio, Weiwei Cheng, Zeno Gantner.

About This Book

In the fast-paced world of fashion e-commerce, the fusion of artificial intelligence and online shopping presents both a wealth of opportunities and exciting challenges. In this manuscript, we present the proceedings of the Fifth Workshop at the Recommender Systems Conference (FashionxRecsys 2023). The proceedings set the stage for a deep dive into the opportunities and challenges in online fashion, drawing insights from a series of innovative studies at the forefront of AI-driven solutions, from both academic and industrial applied science research.

The chapters ahead offer a diverse array of approaches to tackling the complexities encountered in the digital fashion landscape, from automated material analysis to personalized ad creation, each study contributes a unique perspective to addressing the multifaceted hurdles faced by retailers and consumers alike, from beauty product discovery to garment sizing and beyond.

As we navigate through the findings of these pioneering studies, it becomes evident that AI holds a major key to unlocking new possibilities in the world of fashion e-commerce. By leveraging advanced technologies and innovative methodologies, researchers strive to mitigate challenges such as bias in recommendations, fit issues, and consumer segmentation.

Through collaborative efforts and interdisciplinary approaches, we are poised to shape a future where AI not only enhances the online shopping experience but also fosters a more inclusive, personalized, and environmentally conscious fashion ecosystem.

Contents

Size and Fit in Fashion E-Commerce

Tailor: Size Recommendations for High-End Fashion Marketplaces

Alexandre Candeias, Ivo Silva, Vitor Sousa, and José Marcelino

Abstract In the ever-changing and dynamic realm of high-end fashion marketplaces, providing accurate and personalized size recommendations has become a critical aspect. Meeting customer expectations in this regard is not only crucial for ensuring their satisfaction but also plays a pivotal role in driving customer retention, which is a key metric for the success of any fashion retailer. We propose a novel sequence classification approach to address this problem, integrating implicit (*Add2Bag*) and explicit (*ReturnReason*) user signals. Our approach comprises two distinct models: one employs LSTMs to encode the user signals, while the other leverages an Attention mechanism. Our best model outperforms SFNet, improving accuracy by 45.7%. By using *Add2Bag* interactions we increase the user coverage by 24.5% when compared with only using *Orders*. Moreover, we evaluate the models' usability in real-time recommendation scenarios by conducting experiments to measure their latency performance.

1 Introduction

With the increase in online shopping for fashion over the last years [7], customers are more comfortable purchasing garments without any physical contact. In the fashion industry, this leads to an increase in returns related to Size&Fit reasons which bring both financial and environmental losses.

A. Candeias (✉) · I. Silva · V. Sousa · J. Marcelino
Farfetch, Lisboa, Portugal
e-mail: candeiasalexandre@gmail.com

I. Silva
e-mail: ivomiguel.silva@farfetch.com

V. Sousa
e-mail: vitor.costasousa@farfetch.com

J. Marcelino
e-mail: jose.marcelino@farfetch.com

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_1

Different strategies are available to mitigate this problem from an online perspective. Product size-related information available like fitting advice, size tables, and product measurements helps customers when deciding which size to order. However, it lacks customer-specific personalization and size choice is not only dependent on the product but also on the customer preferences. Questionnaires, where the customer inserts data about height, weight, and body shape, can be used to gather customer information and preferences. But it can also be seen as intrusive by customers, and more importantly, this kind of information varies over time and would need to be updated.

Lately, the usage of virtual try-on technologies is also getting some adoption in fashion marketplaces. Even though in clothing articles it still lacks the necessary precision and real feeling to be used more broadly. The customer might also feel intrusive to share image data of his body.

An alternative that requires less customer adoption and can be used in all marketplaces is to use historical interactions between customers and products, for example, past purchases. This data can be used to make specific size recommendations for each pair (customer, product). In this work, we explore how this approach can be used in a luxury fashion marketplace. In the rest of this paper, when we refer to size recommendation we mean the problem of predicting a size for a specific (customer, product) pair.

Luxury fashion marketplaces face some particular challenges when compared to regular fashion marketplaces. The products are premium and have a limited size availability. This makes that a wrong size ordered from one client might represent the end of stock for that specific size of the product. The return process also has some particular issues, due to the high value of the product, returns are more complicated which leads to client satisfaction impacts.

To add complexity, in luxury fashion, the number of products available and brands is usually higher than in normal fashion. This translates into a bigger sparsity of interactions between the user and the products and a bigger number of available sizes.

In this work, we propose "Tailor" a solution to the size recommendation problem in a luxury fashion marketplace that uses only historical transactional data. Our contributions are the following:

1. Findings regarding size recommendations within a high-end fashion marketplace, accompanied by an evaluation of how existing baselines perform when faced with complex data sets.
2. Introduce two models that make use of additional customer signals such as Add2Bag interactions and feedback from returned orders.
3. Analysis of the models' latency, to validate the usability in real-time recommendations scenarios.

2 Related Work

In recent times, there has been a significant surge of academic interest, particularly from the industry, in the Size&Fit domain.

In [6, 13] product-level advising using visual and transactional data is proposed. These approaches can be very efficient in terms of the coverage of users since they don't require a user to have interactions with the marketplace, solving the cold start problem. However, this type of recommendation lacks personalization to the customer-specific needs, and they are often generic such as: "This product usually fits larger than usual" or "This product usually fits smaller than usual".

Virtual try-on targeting Size&Fit customer experience is proposed in [11, 14]. These works focus more on the fitting and visual apparel of the garment in the customer and don't provide any specific size recommendation.

In the size recommendation area, many works have been proposed: [1–5, 9, 12, 16–18]. Bayesian approaches [4, 16], while simple and scalable to large datasets were surpassed by embedding-based models [17]. Embedding-based approaches [1, 2, 17] are usually focused on learning embedding representations for the customer and the product. At inference time, these representations can be used to make predictions however, we are often limited to the customers or products that were used at training time. To overcome the dependency on training time, many sequence-based models have been proposed [3, 5, 12]. These models typically see the user as a sequence of text and use sequence encoder blocks such as transformers to transform that sequence into a fixed-sized representation.

We take a similar approach inspired by [5, 13] which leverages previous purchases as inputs. In our work, we explore how to use other types of user interactions such as Add2Bag to increase user coverage. Besides that, we also introduce feedback from returned purchases.

Including other types of customer data such as weight or height to deal with the problem of cold-start recommendation is studied in [10]. Instead, we focus on increasing the number of information available on the customer's sequence as a way to increase customer coverage.

3 Methodology

We approach the problem of size recommendation as a classification task, where given some user and product information $x = (x_{user}, x_{product})$ we want to predict a probability distribution across the feasible sizes $p(y|x)$. A model $f(x)$ whose output is the distribution $p(y|x)$ can be trained using a classification loss such as Cross-Entropy.

The cardinality of the sizes available in a marketplace can be large, making the sparsity of interactions between users and product variants very high. A product variant is the combination of the pair (product, size). Even though we have a large

number of sizes, they are typically grouped inside a scale in an ordered manner. If we look at the position of a size inside the scale (size position), instead of thousands of sizes we can have dozens of positions. For example, two different sizes "*Gucci EU S*" and "*Gucci IT 46*" share the same relative position (i.e. 1) in their respective scales "*Gucci EU*" and "*Gucci IT*". This means, that instead of predicting a possible size across all the available sizes in the marketplace, we can predict the relative position of the size in the corresponding scale.

To overcome the issue of having recommendations for users that didn't appear at training time, we create a new family of models named Sequence Size Predictor (SSP), these models are inspired by sequence-based recommendation systems [15]. Instead of seeing each user (x_{user}) as a simple *user_id*, we use a sequence of events. You can think of it as $x_{user} = (e_1, e_2, ..., e_T)$, where e_n is a structured textual representation of each event field. This representation is easily transformed into a categorical multivariate time series by encoding each field as integers: $e_n \in \mathbf{R}^d$, where d is the number of fields of each event. The events are user interactions within the marketplace, such as *Order, Add2Bag, Add2Wishlist, etc.*.

Using such a framework is simple to use negative feedback from the users such as returned purchases. We do this by adding a field in the *Order* events named *return_reason*. This field indicates the reason why the order might have been returned, for example: *too large*, *not returned*, or *too small*.

By using such a representation of the user, we are no longer depending on seeing the user in training to make a prediction. Our only requirement is that the user has at least 1 event in the *Event History*. In the next sections, we will explain how to transform the sequence of user events x_{user} in a fixed-size vector representation that can be used in a classification model.

3.1 SSP-LSTM

In the first model, we use a Bi-directional LSTM (Bi-LSTM) to encode the sequence of events, the whole architecture can be seen in Fig. 1.

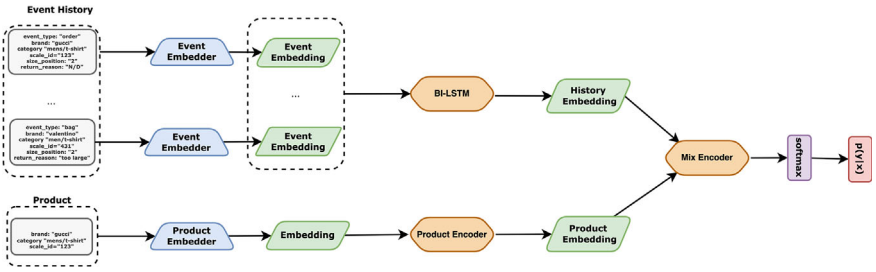


Fig. 1 SSP-LSTM architecture

We start by using an Embedding Layer to transform each of the event fields into a vector representation called "Event Embedding". If we assume that each embedding's field is of dimension v, each event on the sequence is transformed into a vector of size $\#fields \times v$. The sequence of event embeddings is passed through a Bi-LSTM and the last hidden state is kept as the "History Embedding", which is a condensed fixed-size representation of the user. The same Embedding Layer used in the events is also used to transform the common fields present in the product input $x_{product}$. The resulting embedding is passed through a multilayer perceptron (MLP) named "Product Encoder" and its output is used as "Product Embedding". Finally, the "History Embedding" and the "Product Embedding" are concatenated and passed through another MLP "Mix Encoder". Its output is then used as input of a softmax layer to get a valid probability distribution $p(y|x)$.

Architectures of this nature offer several engineering advantages. It becomes feasible to cache the intermediate representation of the "History Embedding" or the "Product embedding", resulting in potential time savings during prediction.

3.2 SSP-Attention

This model is based on a transformer block [19] to encode the event history, it takes inspiration from the attention-based model proposed on [5]. The whole architecture can be found in Fig. 2.

Firstly, we embed the sequence of the events in a similar way to what we described before for the SSP-LSTM model. The only difference is that these embeddings are added a positional encoding based on the number of days that passed since the event occurred. The sequence of "Event Embedding" is then passed through a first transformer block. We use the input sequence as query, key, and value which means that the output will be a sequence of embeddings with the same length as the input, we call this sequence the "Transformed Event History". The same strategy as in the SSP-LSTM is applied to the product input fields which we name "Product Embedding". Notice that in this model we don't apply an extra MLP to the resulting embedding.

To get a fixed-sized vector, we combine the "Transformed Event History" sequence with the "Product Embedding" by using another transformer block. In

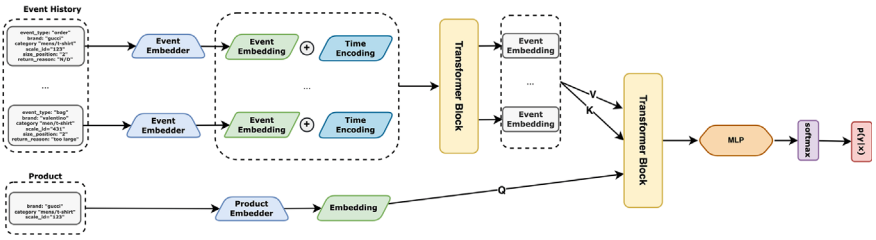


Fig. 2 SSP-attention architecture

this transformer block the key and the value is the "Transformed Event History" sequence and the query is a sequence only with the "Product Embedding", resulting in an output sequence always with length 1. Finally, the output of the last transformer block is passed through an MLP and a softmax layer giving the final output distribution.

4 Experiments

4.1 Baselines

4.1.1 Personalized Most Common Value (PMCV)

The PMCV model is a simple rule-based model, it can be seen as a personalized popularity baseline. This baseline is inspired by the baselines proposed on [3]. The model outputs the most common size for a predefined set of keys seen in the training data. These predefined keys are defined in a hierarchical manner, from more specific (i.e. *user_id, category_id, brand_id*) to less specific (i.e. *brand_id*).

At training time, we save a dictionary containing the most common sizes seen across each key. At inference time, we get the value for the more specific key that is available.

4.1.2 SFNet

SFNet is an embedding-based model proposed in [17]. It learns a representation for each user and another for each product. These representations are jointly encoded using an MLP with many hidden layers, and a final output distribution across the different sizes is computed by using softmax.

To compute the user representation (x_{user}), we use the *user_id* which means we have an embedding vector for each user. To compute the product representation ($x_{product}$), we use the *product_id*, *brand_id*, *category_id*, and the *scale_id*. An embedding is computed for each of the inputs and they are then combined into a single embedding that represents the product by using a MLP.

4.2 Experimental Setting

To train the models, we used historical data of Clothing purchases from our marketplace, excluding returns. We use the size ordered by the user as the label. Notice that even though the training instances don't have return information the event history for the SSP models has events with returned orders, as shown in Figs. 1 and 2.

Our data ranges from 2022-01-01 to 2023-01-01 and contains ≈3.2M purchases from ≈400k different products, and ≈750k different users. We split the data into train/val/test (80%, 10%, 10%) according to a back-testing strategy to maintain causality and prevent data leakage.

SFNet was trained using a batch size of 8128 and Adam optimizer [8] ($lr = 1e{-}3$, $wd = 1e{-}5$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) for 35 epochs with early stopping based on the validation set top1 accuracy.

SSP models were trained using a batch size of 1024 and Adam optimizer ($lr = 1e{-}3$, $wd = 1e{-}4$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) for 25 epochs with early stopping. All SSP models use an event history with a maximum of 20 events, the events present in the history are *Order* and *Add2Bag*.

4.3 Model Performance Comparison

4.3.1 Model Evaluation

We evaluate the models in 4 different scenarios:

- General: no filtering is applied to the test set
- Multiple Gender: test set is filtered to only contain a subset of users that have bought products from multiple genders.
- VIP: test set is filtered to only contain power users, a subset of users that have bought more than the average user in the past.
- Training Users: test set is filtered to only contain a subset of users that were present in the training set.

The results of the different models on these scenarios can be seen in Table 1.

All the models show a better performance than the PMCV baseline, which shows the importance of ML models vs heuristics in the size recommendation problem in a luxury marketplace. The SSP-Attention model performs better than all the other models across all the different scenarios.

The SSP models show better results than SFNet in all the scenarios. However, it's interesting to see that on VIP users the SSP models don't show the same trend as SFNet, behaving worse than in the general scenario. This might be due to the fact that since VIP are power users, they interact more with the marketplace and that might make the need to use a bigger event history.

In all the models, we see a decrease in performance on the MultipleGender vs General scenario. This is due to the fact that these users are more challenging since they have purchased for different genders under the same account, making it more difficult to predict the underlying size of the customer.

Finally, in the Training Users scenario, we see that SSP models still beat SFnet and PMCV. It is also important to note that by filtering the users we are reducing the coverage of users to 49.27% of the users present in the General scenario. Remember that in this scenario we are guaranteed to have an embedding that was specifically

Table 1 Model accuracy in the different scenarios

Model	General			Multiple gender			VIP			Training users		
	top1_acc	top2_acc	top3_acc	top1_acc	top2_acc	top3_acc	top1_acc	top2_acc	top3_acc	top1_acc	top2_acc	top3_acc
PMCV	0.328	0.504	0.606	0.305	0.455	0.535	0.344	0.501	0.574	0.353	0.473	0.517
SFNet	0.387	0.660	0.821	0.378	0.651	0.815	0.425	0.707	0.855	0.449	0.727	0.872
SSP-LSTM	0.538	0.803	0.916	0.479	0.749	0.881	0.489	0.764	0.893	0.523	0.794	0.911
SSP-Attention	0.564	0.815	0.922	0.505	0.762	0.890	0.508	0.776	0.901	0.548	0.808	0.918

tuned to the user at training time, this explains the better performance of SFnet vs the General scenario.

4.3.2 Influence of *Add2Bag* Events

In this section, we see the effect of using *Add2Bag* events in the SSP models. To do this ablation we use a subset of the data that only has users that had both *Add2Bag* and *Order* events in the range of the test set.

We split the usage of the *Add2Bag* events in two different ways: (1) present in the event history at inference time; (2) present in the event history at training time. These two ways can even be mixed, i.e., we can have a model that was trained with *Add2Bag* events in the event history and at inference time also has *Add2Bag* events in the event history of the instances for which is predicting. In Table 2, you can see the results for the SSP models for all 4 different combinations:

1. *Add2Bag* in event history and trained only with *Order* in event history.
2. *Add2Bag* in event history and trained with *Order* and *Add2Bag* in event history.
3. no *Add2Bag* in event history and trained only with *Order* in event history.
4. no *Add2Bag* in event history and trained with *Order* and *Add2Bag* in event history.

For both models, we see that the best results are when we use *Add2Bag* and *Order* events in the event history at training time and at inference time.

We can also see that for the SSP-LSTM model if we don't have *Add2Bag* events at training time and only use them at inference we have worse results. In the SSP-Attention model, we see that we can have good accuracy by only adding *Add2Bag* events at inference time.

Besides this accuracy improvement, by using *Add2Bag* events we see an increase of 24.5% in the users that can receive a size recommendation. Remember that to make a recommendation our requirement is that the user has at least 1 event in history.

Table 2 Effect of *Add2Bag* events in SSP models

Model	Add2Bag in event history	Trained only with order events in history			Trained with Add2Bag and orders events in history		
		top1_acc	top2_acc	top3_acc	top1_acc	top2_acc	top3_acc
SSP-LSTM	Yes	0.451	0.738	0.887	0.528	0.797	0.913
SSP-Attention	Yes	0.550	0.809	0.919	**0.554**	**0.810**	**0.919**
SSP-LSTM	No	0.496	0.771	0.901	0.498	0.772	0.901
SSP-Attention	No	0.515	0.780	0.904	0.513	0.779	0.904

Table 3 Effect of using *return_reason*

Model	return_reason	top1_acc	top2_acc	top3_acc
SSP-LSTM	Yes	0.497	0.771	0.901
SSP-Attention	Yes	**0.515**	**0.781**	**0.905**
SSP-LSTM	No	0.494	0.767	0.898
SSP-Attention	No	0.503	0.769	0.898

4.3.3 Influence of *return_reason* Field

We present in Table 3 an ablation regarding the effect of using or not using the *return_reason* field in the SSP models. This ablation was done using a dataset only with *Order* events in the event history.

We see that for the SSP-Attention model, we have an increase of 2.33% in accuracy, while in the SSP-LSTM model the increase is negligible.

4.4 Model Latency Comparison

In a real production use case, an essential performance metric to consider is model latency, which measures the time the model takes to make a prediction (or a batch of predictions) in milliseconds.

Figure 3 illustrates how the average model latency changes with the batch size and the length of the event history using synthetic data and models with consistent hyperparameters as described in the Experimental Setting section. The inference is performed on a CPU (*11th Gen Intel® Core™ i7-11800 H @ 2.30 GHz × 16*), which better reflects production machine types.

Figure 3a shows that SFNet consistently exhibits lower latency than the SSP models across all batch sizes, primarily because SFNet processes smaller inputs (without sequences of events) and has fewer computations. For the SSP models, only results with 20 events in the history are presented.

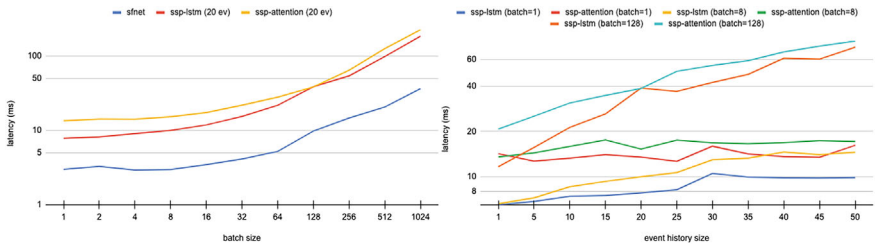


Fig. 3 Model latency benchmark

The comparison between SSP-LSTM and SSP-Attention reveals that SSP-LSTM consistently has lower latency due to the extra computation involved in the 2 attention layers of SSP-Attention. Both models achieve a latency of under 50ms for batch sizes smaller than 128. In production, a batch size of 1 is commonly used, and all models have a latency well below 20ms, respectively 2.99 ms for SFNet, 7.81 ms for SSP-LSTM, and 13.51 ms for SSP-Attention.

Figure 3b examines how model latency changes with the size of the event history for various batch sizes. For larger batch sizes (128), latency increases with an increase in event history size. Interestingly, latency appears to stabilize for smaller batch sizes after 30 events in history. The insight here is that this behavior is possibly linked to torch optimization, mainly when batching a limited number of events when making fewer predictions (batch size = 1 or 8).

5 Conclusion

In this work, we showed how size recommendations can be solved in a luxury fashion marketplace using only historical purchase data.

By using SSP models with an attention mechanism (SSP-Attention), we achieve an improvement of 72.2% in top1 accuracy compared to a rule-based heuristic, 45.7% compared to SFNet [17] and 4.8% compared to SSP-LSTM. We also show the importance of using *Add2Bag* interactions to increase both the accuracy and coverage of the customers. Finally, we show that SSP models besides increasing the accuracy are still capable of having a latency below 15ms which is still usable in production.

For future work, we would like to explore ways of improving the SSP models by including more information regarding the product, such as product measurements. Including more information regarding the customer such as measurements, body shape, and other implicit signals besides Add2Bag is also an open topic.

6 Appendix

6.1 PMCV Hyper-Parameters

```
input_hierarchy:
  - [''user_id'', ''category_id'', ''brand_id'']
  - [''user_id'', ''cateogry_id'']
  - [''user_id'', ''brand_id'']
  - [''user_id'']
  - [''category_id'', ''brand_id'']
  - [''category_id'']
```

6.2 SFNet Hyper-Parameters

```
user_inputs:
    – ‘‘user_id’’,
product_inputs:
    – ‘‘brand_id’’
    – ‘‘category_id’’
    – ‘‘scale_id’’
    – ‘‘product_id’’
embedding_dim: 20
user_pathway_dims: [50, 30, 20]
item_pathway_dims: [50, 30, 20]
combined_pathway_dims: [100, 200, 400, 1000]
activation: ‘‘tanh’’
dropout: 0.05
```

6.3 SSP-LSTM Hyper-Parameters

```
embeddings_parameters:
  brand_id:
    dim: 8
  category_id:
    dim: 4
  return_reason:
    dim: 2
  size_position:
    dim: 3
  scale_id:
    dim: 6
  event_type:
    dim: 2
BiLSTM:
  dropout: 0.01
  hidden_layer_size: 25
  number_layers: 2
ProductEncoder:
  activation: relu
  dropout: 0.01
  hidden_state_dimensions:
    – 50
    – 25
MixerEncoder:
  activation: relu
  dropout: 0.01
  hidden_state_dimensions:
    – 150
    – 100
```

6.4 SSP-Attention Hyper-Parameters

```
embeddings_parameters:
  brand_id:
    dim: 64
  category_id:
    dim: 48
  return_reason:
    dim: 2
  size_position:
    dim: 16
  scale_id:
    dim: 48
  event_type:
    dim: 2
TransformerBlock1:
  num_heads: 12
TransformerBlock2:
  num_heads: 16
MLP:
  activation: relu
  dropout: 0.01
  hidden_state_dimensions:
    – 150
    – 100
```

References

1. Abdulla G, Borar S (2017) Size recommendation system for fashion E-commerce. KDD workshop on machine learning meets fashion 2017:1–7
2. Dogani K, Tomassetti M, De Cnudde S, Vargas S, Chamberlain B (2019) Learning embeddings for product size recommendations. In: CEUR workshop proceedings, p 2410
3. Eshel Y, Levi O, Roitman H, Nus A (2021) PreSizE: predicting size in e-commerce using transformers, vol 1. Association for Computing Machinery
4. Guigourès R, Sheikh AS, Ho YK, Bergmann U, Koriagin E, Shirvany R (2018) A hierarchical Bayesian model for size recommendation in fashion. In: RecSys 2018 - 12th ACM conference on recommender systems, pp 392–396
5. Hajjar K, Lasserre J, Zhao A, Shirvany R (2021) Attention gets you the right size and fit in fashion. In: Dokoohaki N, Jaradat S, Corona Pampín HJ, Shirvany R (eds) Recommender systems in fashion and retail. Springer International Publishing, Cham, pp 77–98
6. Karessli N, Guigoures R, Shirvany R (2019) SizeNet: weakly supervised learning of visual size and fit in fashion images. In: IEEE computer society conference on computer vision and pattern recognition workshops, pp 335–343
7. Keenan M (2022) The state of the ecommerce fashion industry: statistics, trends & strategies to use in 2023. Accessed 1 July 2023
8. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Bengio Y, LeCun Y (eds) 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, conference track proceedings
9. Lasserre J, Sheikh AS, Koriagin E, Bergman U, Vollgraf R, Shirvany R (2020) Meta-learning for size and fit recommendation in fashion. In: Proceedings of the 2020 SIAM international conference on data mining, SDM 2020, pp 55–63

10. Lefakis L, Koriagin E, Lasserre JA, Shirvany R (2020) Personalized size recommendations with human in the loop. In: Proceedings of the 37th international conference on machine learning, Vienna, Austria
11. Lewis KM, Guttag J (2022) SizeGAN: improving size representation in clothing catalogs
12. Mohammed Abdulla G, Singh S, Borar S (2019) Shop your right size: a system for recommending sizes for fashion products. In: The web conference 2019 - companion of the world wide web conference. WWW 2019, pp 327–334
13. Nestler A, Karessli N, Hajjar K, Weffer R, Shirvany R (2021) SizeFlags: reducing size and fit related returns in fashion E-commerce, vol 1. Association for Computing Machinery
14. Pecenakova S, Karessli N, Shirvany R (2022) Fitgan: fit- and shape-realistic generative adversarial networks for fashion. 2022 26th international conference on pattern recognition (ICPR). IEEE Computer Society, Los Alamitos, CA, USA, pp 3097–3104
15. Quadrana M, Jannach D, Cremonesi P (2019) Tutorial: sequence-aware recommender systems. In: The web conference 2019 - companion of the world wide web conference, WWW 2019, vol 1(1), p 1316
16. Sembium V, Rastogi R, Tekumalla L, Saroop A (2018) Bayesian models for product size recommendations. In: The web conference 2018 - proceedings of the world wide web conference, WWW 2018, pp 679–687
17. Sheikh AS, Ho YK, Guigourès R, Shirvany R, Bergmann U, Koriagin E, Vollgraf R (2019) A deep learning system for predicting size and fit in fashion e-commerce. In: RecSys 2019—13th ACM conference on recommender systems, pp 110–118
18. Singh S, Abdulla GM, Borar S, Arora S (2018) Footwear size recommendation system. AI meets Fashion workshop, KDD
19. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Lu, Polosukhin I (2017) Attention is all you need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds) Advances in neural information processing systems, vol 30. Curran Associates Inc

Exploring Fit and Shape for Predicting Garment Size Issues in Fashion with Multi-task Learning

Sahar Mbarek, Leonidas Lefakis, Peter Gehler, Sonia Aurelio, Rodrigo Weffer, and Reza Shirvany

Abstract Online shoppers face many difficulties finding the right garment size that fits them. There has been a large body of work that try to tackle this problem from a customer perspective. Most often the approaches rely solely on purchases and returns to offer a personalized size-recommendation. In this work we investigate the potential effect of additional fashion characteristics such as shape and fit attributes of an article to better predict a potential size issue even before a purchase has been made. We frame the problem as a Multi-Task-Learning (MTL) problem and extend prior work to predict size issues from garments visual information. In the experiments, we explore different MTL architectures and a variety of datasets to further harness the power of these models. To the best of our knowledge, the proposed approach is the first to combine size, fit and shape attributes of fashion articles to address the challenge of size issue prediction in online shopping. Our findings indicate that there is a positive effect of additional attributes and that we can improve the size-issue prediction in comparison to the state-of-the-art models.

S. Mbarek (✉) · L. Lefakis · P. Gehler · S. Aurelio · R. Weffer · R. Shirvany
Zalando SE, Tamara-Danz-Str. 1, 10243 Berlin, Germany
e-mail: sahar.mbarek@zalando.de

L. Lefakis
e-mail: leonidas.lefakis@zalando.de

P. Gehler
e-mail: peter.gehler@zalando.de

S. Aurelio
e-mail: sonia.aurelio@zalando.de

R. Weffer
e-mail: rodrigo.weffer@zalando.de

R. Shirvany
e-mail: reza.shirvany@zalando.de

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_2

1 Introduction

When buying clothes online, after a customer has identified a garment of their liking, they need to select a size they want to purchase the garment in. To support this decision, size information is usually part of the garment information, for example in the common Alpha size range (XS, S, M, L, etc.) or Numerical sizes (36, 38, 40, etc.). It is only after the garment arrives in customers' hands that they can finally try it on and discover whether their expectation of the nominal size they have selected days ago matches the real fit of the garment for them. In this work, when the nominal size does not match the actual real world size and fit of the garment we consider there is a *size issue* instance. In other words, if the garment does not fit in its size as it is supposed to, we consider this a size issue and we aim to build a model that predicts this early on even before the customers purchase the garments. Non-fitting garments are usually returned and lead to customer dissatisfaction, negative environmental impact, and additional costs. Furthermore, the delayed verification of the garment fit typically discourages customers to engage with new brands and items they are not familiar with. Size prediction is intrinsically challenging. The main contributing factors are (a) a coarse definition of size systems (b) brand specific size specifications (e.g. Small in brand A could be Medium in brand B) and (c) the inability to convey the overall garment fit in a single number or letter. In addition, sizes are hard to convert to each other, so in order to tackle this challenge fully, a model capable of delivering a per-garment size advice should be the goal. One source of information for size-advice is a purchase history or return-rates of garments. Predicting size issues before any purchases has been addressed before, e.g. [1] with the use of return-rates of historical purchase data as a training signal and with product images as the sole input. To the best of our knowledge we are the first to investigate whether additional characteristics of the garments such as expert annotations of their fit (loose, tight) and shape (straight, tapered) attributes can yield improved size issue prediction performance. Of course, predicting size from images is only one piece of a broader solution, a production system would combine multiple sources, such as physical measurements, customer purchase history and information about the sizing system used by the brand, e.g. [2]. Once we have identified a potential size issue, that is, an item "runs small" or "runs large", this finding can be shown to the customer as a recommendation to buy a size larger or smaller compared to the one they would normally choose. A successful system leads to better size advice, and thus to significantly lower return rates and higher customer satisfaction [2]. The system we propose in this paper extends the work of SizeNet [1]. SizeNet is a Multi-Layer-Perceptron (MLP) that predicts from the FashionDNA [3] image representation of a product image whether an article has a size issue or not. A binary size-issue is defined as a deviation of the item return rate from the expected category return rate. Our work is the first to mix fit and shape characteristics to predict a potential size issue. We explore different architectural solutions capable of leveraging a multi-task learning setting to not only predict a "size issue" (as has been proposed in the past) but in addition the very heart of the challenge, that of the human judgements of fit and shape. To enable training we collect a novel

dataset with varying degree of supervision so we can study the effect of data set size in addition to model choices. Our experimental results with real world data and the follow up rigorous analysis shows that multi-task learning performs comparably to size-issue prediction, however the effect is more pronounced in smaller data regimes. Through our experiments we also confirm that dataset size and network size have a strong effect on model performance and our final model drastically outperforms the state-of-the-art SizeNet [1] performance due to the compounded effect of all three factors: (a) dataset size, (b) model size, and (c) multi-task learning.

2 Related Work

2.1 Size Advice

The absence of physical try-ons and the variation in sizing charts across different brands often lead to uncertainty and dissatisfaction in finding the right size when shopping online. There has been several work dedicated to addressing the various aspects of this problem [1, 2, 4–15]. One category of approaches is to use customer purchase history to provide a personalized size recommendation [7–9, 13–15]. In this line of work, [13] proposed a deep learning based content-collaborative method for personalized size and fit prediction, where customer and articles features are embedded into a latent space to learn the target size&fit. Dogani et al. [16] proposed a size recommendation through a product size based latent space. Lasserre et al. [14] developed a meta-learning deep learning method for size recommendation, that exploits large data scale and absorbs new data points efficiently without the need for re-training. A recent work [15] used an attention-based model to encode customer history to corresponding sizes. In another line of work [1, 2], the size and fit problem is tackled based on article specific information. Customers are then provided by article based size advice such as "*article runs too small or too big*". The authors propose a Bayesian model to predict size issue (too-big, too-small, true-to-size) by exploitation of article size returns and prior information from experts on garment size. However, a sufficient amount of orders and returns is required to have certain confidence level in the predictions. This is complicated by the fact that in fashion e-commerce returns can only be recorded with a certain delay. In order to overcome the cold-start-problem, [2] exploited, in addition to human experts, a *SizeNet* model as prior to the Bayesian model. The author of SizeNet [1] suggested using visual cues from article images to predict size issue. This solves the cold-start-problem to be able to raise potential size issue even before any return is observed.

2.1.1 Multi-task Learning in DNN

Multi-task learning [17] is a machine learning paradigm that aims to simultaneously solve multiple tasks, e.g. classification tasks using shared representation with the goal of enhanced generalization and better performance because of information sharing across tasks. In the field of computer vision, MTL has showed promising results in learning multiple tasks using a shared convolutional network based model [18–21], while minimizing negative transfer between tasks. The existing MTL architectures have often been partitioned in two main categories, those with *hard parameter sharing* (HPS) and those with *soft parameter sharing* [22] (SPS). Traditionally, a hard-parameter sharing MTL architecture is composed of a global feature extractor, usually made of convolutional layers, followed by task specific output branches [23]. [24] proposed Multi-task network cascades to learn semantic segmentation, where the output of each task specific branch is appended to the input of the next learned task, forming the cascade of information flow. Liu et al. [25] suggested a Multi-task attention network that uses a single shared network with a soft-attention module for each task. Recently, [26] introduced Multi-gate Mixture-of-Experts, which consists of multiple shared models (experts) and task-specific output heads. For soft parameter sharing MTL architectures, each task has a separate network and the information flows between parallel layers modeled using a learned linear combination of the outputs of each task, e.g., Cross-Stitch unit [27]. Later, [28] generalized Cross-Stitch further by proposing to exploit a 1×1 convolutional layer to combine output features from different task layers.

3 Datasets

To investigate the main research question of our work, whether size, fit and shape information offer complementary information, we collected a new dataset of women textile garments, detailed in this section and summarized in Table 1. We also make use of dataset augmentation techniques with extra annotations to enable exploration of a MTL setup as follows.

Table 1 Dataset statistics of MTLD and its subsets. For all items we have size-issue labels

Dataset	#Articles	% Fit annotations (%)	% Shape annotations (%)	%Fit+Shape annotations (%)
MTLD	340,015	83	42	24
MTLD-fit	280,849	100	30	30
MTLD-shape	141,678	59	100	59
MTLD-small	82,512	100	100	100

3.1 MTLD: A Size Issue Dataset

The garments come from 12 categories (dresses, jeans, jersey, etc.) and about 1000 different brands. When it comes to articles that only differ in color, we treat those as distinct samples for the size issue. This choice has been motivated by expert advice and is further justified by the fact that the fabric of the garment is different depending on dying technique, which leads to different size and fit characteristics of the garment and thus different return reasons. We create the dataset in a way that it is balanced across size issues, i.e. it has about the same number of size-issue and no-size-issue items. Every sample in the dataset is a tuple $(x, y_{size}, w, y_{fit}, y_{shape})$ with $x \in \mathbb{R}^{128}$ being the FashionDNA [3] image representation, $y_{size} \in \{0, 1\}$ the size issue label, $w \in \mathbb{R}_+$ (y_{size} and w come from the teacher model of SizeNet which will be described in Sect. 4.1), $y_{fit} \in \{0, \ldots, 5\}$, the fit label, and the shape label $y_{shape} \in \{0, \ldots, 6\}$. We refer to this dataset as MTLD. It contains a total of 340,015 items, of which 183,967 (54%) have size issues and 156,048 (46%) do not.

3.2 MTLD-{Fit,Shape,Small}: Adding Fit and Shape Annotations

Both fit and shape are important factors to determine the size and style of a garment. Fit describes the width of a garment in relation to the wearer's body. We consulted with fashion experts who have composed the following five different classes of fit ordered from closest to the body to furthest $y_{fit} \in \{skinny, slim, regular, loose/relaxed, oversized\}$. The dataset includes more than 280k items with fit information. The shape then describes the intended silhouette of a garment in relation to the wearer's body and it is classified in six categories $y_{shape} \in \{fitted, straight, flared, tapered, cocoon, body\}$—*hugging*. Shape annotation is available for about 140k items. Representative examples are shown in Fig. 1. In fashion industry, in practice the annotation of the fit and shape attributes comes from different sources with varying accuracy, including fashion-experts, brands, garment fitting experts, Brands, Data sheets, etc., which leads of to have have "full" annotations for more than 82k items albeit with different levels of accuracy in the annotations based on their sources. In Fig. 2 we show the available fit and shape data statistics along with the percentage of items having a size issue.

We define subsets of MTLD depending on the annotations available, MTLD-fit is the MTLD subset for which we have also fit attributes, and MTLD-shape the subset with shape annotations. The smallest dataset is MTLD-small that contains only items for which we have all label information available.

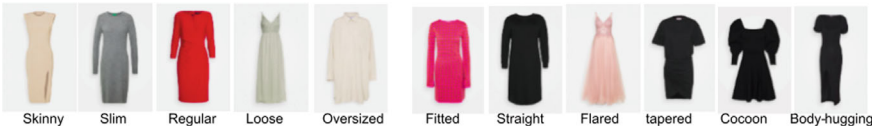


Fig. 1 Representative examples of the five fit (left) and six shape (right) categories

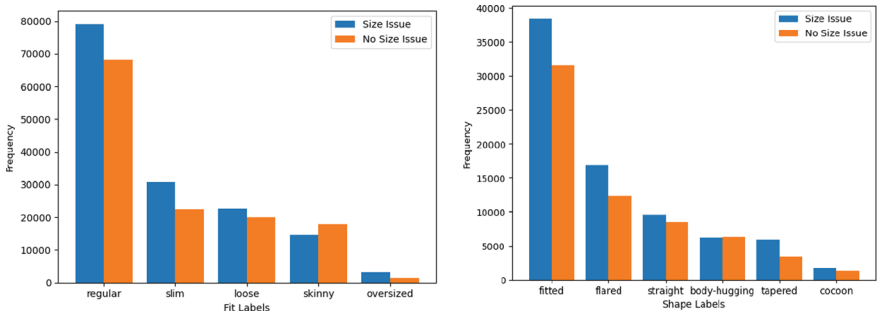


Fig. 2 Distribution of fit/shape labels in MTLD

4 Methods

We first review the SizeNet model of [1] that serves as a baseline starting point for our work and then describe multiple MTL variants that we tested to mix size, fit and shape information. We acknowledge that the list of considered MTL methods is not exhaustive and that other network designs are conceivable. However, we believe the list is sufficient to empirically validate our main question on the added effect of fit and shape information to predicting size issues. All models leverage the 128-dimensional FashionDNA embedding [3] vector of the extended FashionDNA network whose parameters are fixed and not updated. It has been thoroughly shown that one can also use any other available pre-trained backbone model instead of the FashionDNA, albeit with some potential degradation of the final results [1].

4.1 SizeNet

The work of [1] proposes a weakly-supervised teacher-student framework with the goal to predict a size issue given the images of the fashion garments. The teacher is a statistical model that turns historical data of sold articles and returns into a binary size issue flag and a confidence score about this decision. The SizeNet student is then a four layer MLP that predicts size issue from an image of the garment alone. Both our work and [1] use the convolutional neural network embedding FashionDNA [3] as an image backbone. When training the MLP parameters we use a 0.5 dropout

and ReLU activation function. The size issue prediction task is defined as a binary classification task. We consider an article to either have a size issue (class +1, i.e., the article runs too small or too big) or being true-to-size, (class 0, no size issue). This is determined when comparing the size related returns to the average of what we would expect in the fashion category $y_{category}$ the item is a part of, over the same sales period. For every article we have further information: (1) its fashion category $y_{category}$, e.g., t-shirts, dresses, trousers, etc., and (2) its sales period: depending on the period of time, the size return rate and customer behavior in a fashion category might differ. The teacher also computes a confidence score s, that depends on the number n of sold items for the given article, the observed returns k and the expected return rate probability of the category $y_{category}$ it belongs to. The binomial likelihood of observing k size-related returns out of n orders with a size-related return rate probability p is

$$\mathcal{L}(n, k, p) = \binom{n}{k} p^k (1-p)^{n-k}, \tag{1}$$

from which we compute the per-item (n, k are item specific) confidence score as the negative log-likelihood $s = -\ln(\mathcal{L})$.

The confidence score s is high when the observed return rate k/n deviates from the category mean p, e.g. if the size related return is very high (size issue, class 1), or very low (no size issue, class 0). The former is used to weight each training sample. In order to teach the student-SizeNet about the low confidence of some labels, each training sample is assigned a weight $w = \ln(1 + s)$. The logarithmic transformation is applied to the score s to reduce the skewness in the confidence score distribution and for numerical stability, see [1] for more details on this setup.

4.2 Multi-task Learning

We now turn to multi-task formulations, to explore the joint prediction of size, fit and shape. The main idea of multi-task learning is that predicting multiple tasks together is easier than predicting in isolation. Here we consider the different available annotations y_{size}, y_{shape}, and y_{fit} as the different tasks where the first is a binary classification task and the latter two a categorical prediction. In the remainder of the section we explain MTL architectures that have different ways of parameter sharing.

4.2.1 Hard Parameter Sharing

Hard Parameter Sharing (HPS) stands out as a prominent paradigm in MTL, characterized by a global features extractor, typically a Convolutional Neural Network, coupled with distinct task-specific layers, termed branches, to capture individual task characteristics. In Fig. 3 we depict the network layout, every task has the same layout as SizeNet. A common two base layers are shared for all three

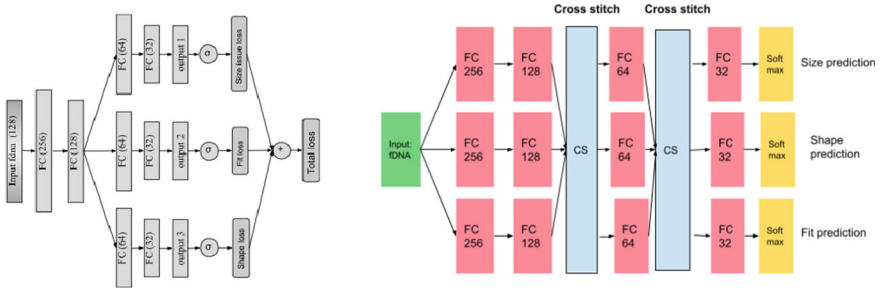


Fig. 3 The two main MTL architectures, hard parameter sharing (left) and cross-stitch (right)

tasks as is the FashionDNA image embedding. Then separate branches predict the binary and categorical variables respectively. The HPS model is trained on a joint loss function that is the weighted linear combination of task-specific losses, $Loss_{total} = \alpha_{fit} Loss_{fit} + \alpha_{shape} Loss_{shape} + \alpha_{sizeissue} Loss_{sizeissue}$. For items with missing annotations we cannot compute the loss and propagate the gradients. We train the models using stochastic gradient descent, the learning rate is found via careful cross validation. We test the following hard parameter sharing MTL models. **MTL-Fit**: Two-task model to predict size and fit; **MTL-Shape**: Two-task model to predict size and shape; **MTL-{Fit,Shape}**: Three-task model to predict size, fit and shape. Since above models introduce more model parameters and we want to disentangle the model capacity effect, we test the following SizeNet variants where we scale every layer to achieve similar parameter numbers as for the HPS models.

- **SizeNet-medium**: SizeNet with layer sizes (256-144-96-64-2) to match the two-task models.
- **SizeNet-large**: SizeNet with layer sizes (512-256-128-64-2) to match the three-task models.

4.2.2 Cross-Stitch Architecture

Cross-stitch [27] is a common technique in MTL that falls into the category of soft-parameter sharing. It includes cross-stitch units into a network architecture, and enables information flow between shared layers across multiple tasks. Formally, a cross-stitch unit C is a linear combination of the outputs of given hidden layers from every task. For example, if we consider the output of three hidden layers h_1, h_2, and h_3 from tasks $size$, fit, and $shape$, respectively the outputs of the cross units for the three tasks is the following (where h are vectors and c_{ij} scalars): $\begin{bmatrix} \tilde{h_1} \\ \tilde{h_2} \\ \tilde{h_3} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}$. The network can learn to make certain layers task specific by setting shared values inside C to zero or linearly blend them. The cross-stitch are applied to the output of different fully connected layers. In this work, we use a

Cross Stitch model to simultaneously learn all three tasks and do not design two-task versions. The model layout is shown on the right in Fig. 3, we use a SizeNet-MLP per task with additional cross-stitch layers. The original work [27] suggested a two-step training, first each task separately, then fine-tuning with cross-stitch units introduced. This strategy lead to overfitting in our experiments, therefore we opted to learn all parameters including the cross-stitch layer jointly from scratch. We further experimented with the layout and where to put the cross-stitch units, the model shown on the right in Fig. 3 worked best.

5 Experiments

Our new dataset allows us to investigate our initial question on whether additional label information does aid the performance when presented in a MTL framework. Further we investigate other contributing factors of performance such as dataset and model size. The performance metric is Average Precision (AP) of the size issue binary prediction, which summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold. We also report the Area under the ROC (AUC) as an additional metric.

For all models we evaluate the hyper-parameters using a Monte Carlo cross-validation scheme. Initially we split into 80% train and 10% validation to select the best hyper-parameters such as learning rate, batch size, number of training epochs and loss factors. We use the AP of size issue as a metric to select the best model. With these found and fixed hyperparameters we then train models on 100 splits of the data leaving 20k items for testing every time. The reported results are the average scores along with the confidence intervals. We distinguish three different model regimes depending on their parameter count and compare results only across similar sized model. SizeNet is the smallest model, followed by MTL-Shape, MTLD-fit, MTL-{Fit,Shape}, and SizeNet-medium with a comparable number of parameters. The largest models are SizeNet-large and Cross-stitch. The experimental results are shown in Fig. 4 for all datasets and for MTLD also listed in Table 2.

5.1 The Effect of More Model Parameters

Larger models perform better, we find a strict ordering of model performance from SizeNet, SizeNet-medium, to SizeNet-large models consistent for all dataset sizes (diamond vs. triangle makers in Fig. 4). Cross-stitch and SizeNet-large are the best performing models on the largest dataset MTLD (diamond markers in top-right corner of Fig. 4). This indicates that the models can be further increased, as an avenue to increase absolute performance in future work.

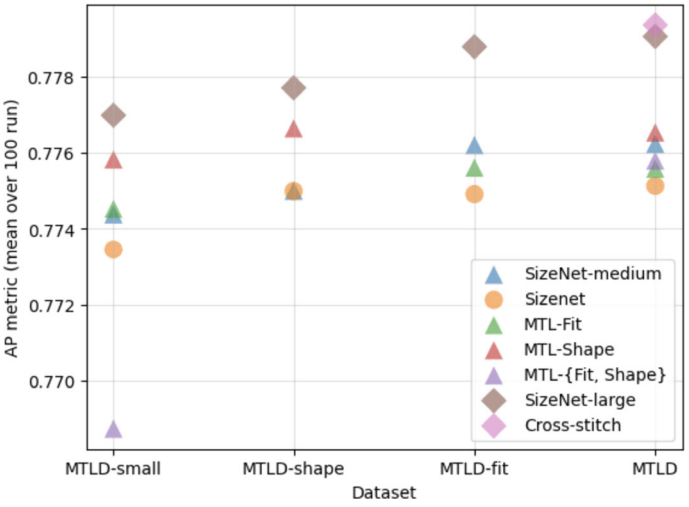


Fig. 4 AP for different training splits (increasing from left to right) and different architectures. Models with the about the same number of parameters share the same marker

Table 2 Parameter count and performance with confidence intervals [29] of different variants trained on MTLD. From top to bottom the models are ordered by parameter count from small to large

Variant	#Params	AP	AUC
SizeNet	76,322	0.7751[0.7679, 0.7805]	0.74920[0.7443, 0.7548]
SizeNet-medium	90,290	0.7762[0.7688, 0.7814]	0.7504[0.7455, 0.7548]
MTL-Fit	82,727	0.7755[0.7681, 0.7813]	0.7496[0.7451, 0.7549]
MTL-Shape	86,856	0.7765[0.7685, 0.7817]	0.7504[0.7449, 0.7563]
MTL-{Fit,Shape}	97,357	0.7757[0.7687, 0.7818]	0.7499[0.7453, 0.7549]
SizeNet-large	**238,658**	0.7790[0.7718, 0.7840]	**0.7532[0.7485, 0.7581]**
Cross-stitch	229,215	**0.7793[0.7720, 0.7856]**	0.7531[0.7475, 0.7589]

5.2 The Effect of More Data

The second finding in line with prior expectation is that model performance increases with more data. In Fig. 4 the dataset size is ordered from small (left) to large (right) and we see corresponding models increase in performance rather consistently. Both MTL-Shape and SizeNet-medium appear to be levelling off (no increase from 2nd column MTLD-shape to 4th column MTLD), while the largest model SizeNet-large (brown diamond marker) still benefits from a larger training set and achieves best performance on MTLD. The higher increase for SizeNet-medium and SizeNet-large indicates that the networks are not saturated yet and would benefit from more data and model parameters.

5.3 The Effect of Fit and Shape on Size Prediction

We now turn our attention to the main hypothesis, that is, modeling fit and shape tasks jointly with predicting size issues benefits performance. For the two-task model MTL-Shape we find a better performance than the same-size SizeNet-medium (red triangle is higher than blue triangle for all datasets). This is not true for MTL-Fit, while on MTLD-small the performance is on par with SizeNet-medium it falls a bit behind on larger datasets (green triangle slightly lower than blue triangle). Adding the shape task does improve performance over a baseline, however the effect shrinks as dataset grows (red triangle from left to right column). Fit information appears to have no additional effect on size performance. This may also explain the next finding: the three task model MTL-{Fit,Shape} does not improve over its two-task counterparts, it is of slightly lower performance that than the comparable SizeNet-medium on MTLD. The low MTL-{Fit,Shape} performance on the small MTLD-small (purple diamond, left column) is likely an optimization problem and a more exhaustive hyper-parameter search may increase the performance to a similar level as the other methods. The overall best performing MTL model is Cross-stitch it is slightly higher than the single task SizeNet-large but with the added benefit of also being able to predict fit and shape.

5.4 Size Issue Performance Across Fit/Shape

In order to explore in-depth the effect of including fit and shape, we compare the performance of predicting the size issue of Cross-stitch and SizeNet across fit and shape (Figs. 5 and 6). We train and test the model on the same data MTLD. The cross stitch model outperforms the SizeNet in all shape categories, except the cocoon one (-0.0096pp). This can be explained by the fact that we have the least data for this category of shape. We also have looked to the common failures of SizeNet and Cross-stitch across the shape attribute. We observe that some of the common failures might have poor data annotation for the shape attribute. For the fit attribute, Cross-stitch is outperforming baseline SizeNet except in category fit oversized. The improvement across the fit categories is rather minimal 0.005pp. Similarly to the shape, we observed that the common failing cases of SizeNet and Cross-stitch might originate from poor annotation of the fit attribute.

6 Conclusion

Early experimentations and insights were presented in exploring the use of garment attributes such as fit and shape to predict potential size problems. The challenge was tackled from the angle of MTL frameworks, and a novel dataset preparation and

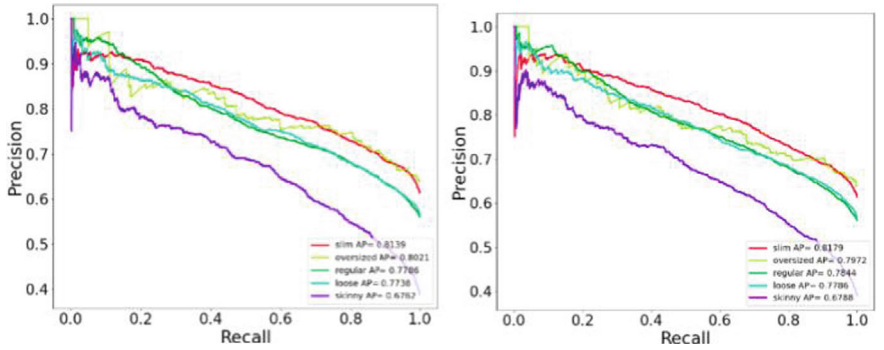


Fig. 5 Precision-recall curve of size issue prediction across shape. Left plot: baseline SizeNet, Right plot Cross-stitch

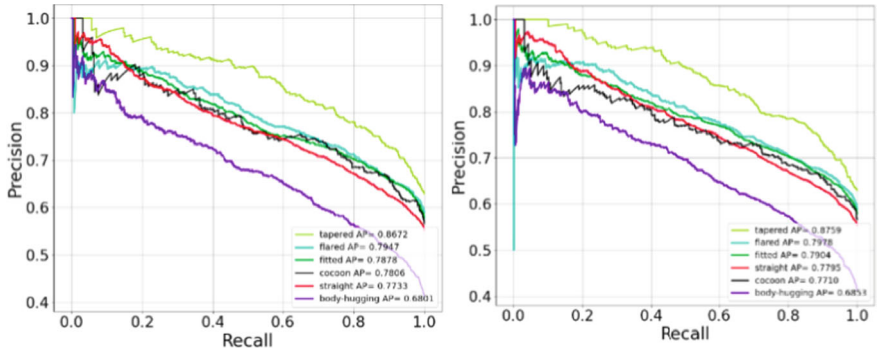


Fig. 6 Precision-recall curve of size issue prediction across fit. Left plot: baseline SizeNet, Right plot Cross-stitch

attribute consideration was proposed to allow critical early experimentations in this domain. Our experimental results indicate that there is a positive effect of additional expert shape but not fit annotation. The effect is more pronounced in smaller data regimes but consistent across different subsets of MTLD. We further confirm that model size has a strong effect of network performance and with SizeNet-large we have identified a model that outperforms previous state-of-the-art which indicates further potential for performance improvements are possible. The reported results suggest that a more in-depth study of the interplay between size on the on hand and shape and fit on the other hand is advised. Possible avenues of future work are to identify subsets of predictive fit and shape categories and to search for segments in the item set where a stronger effect can be found. Further we want to note that while size information is only available after items are sold, shape and fit annotations are possible to obtain before selling so we could design a model that makes use of all available annotation at prediction time. We started with the question on whether fit and shape information contributes to predicting size issues, our findings are nuanced

but we understand them as evidence that this research question is worthwhile to investigate in the future.

References

1. Karessli N, Guigourès R, Shirvany R (2019) In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 0–0
2. Nestler A, Karessli N, Hajjar K, Weffer R, Shirvany R (2021) Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp 3432–3440
3. Bracher C, Heinz S, Vollgraf R (2016) arXiv:1609.02489
4. Ashdown S (2007) Sizing in clothing. Elsevier
5. Diggins MA, Chen C, Chen J (2016) Analytical modeling research in fashion business, pp 31–48
6. Han X, Wu Z, Wu Z, Yu R, Davis LS (2018) Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7543–7552
7. Sembium V, Rastogi R, Saroop A, Merugu S (2017) Proceedings of the eleventh ACM conference on recommender systems. ACM, pp 243–250
8. Sembium V, Rastogi R, Tekumalla L, Saroop A (2018) Proceedings of the 2018 world wide web conference. WWW '18, pp 679–687
9. Guigourès R, King Ho Y, Koriagin E, Sheikh AS, Bergmann U, Shirvany R (2018), pp 392–396. https://doi.org/10.1145/3240323.3240388
10. Abdulla GM, Borar S (2017) KDD workshop on machine learning meets fashion. https://kddfashion2017.mybluemix.net/final_submissions/ML4Fashion_paper_8.pdf
11. Dogani K, Tomassetti M, De Cnudde S, Vargas S, Chamberlain B (2019) SIGIR eCom, Paris, France. https://sigir-ecom.github.io/ecom19Papers/paper13.pdf
12. Misra R, Wan M (2018). J McAuley. https://doi.org/10.1145/3240323.3240398
13. Sheikh AS, Guigourès R, Koriagin E, Ho YK, Shirvany R, Vollgraf R, Bergmann U (2019) Proceedings of the 13th ACM conference on recommender systems, pp 110–118
14. Lasserre J, Sheikh AS, Koriagin E, Bergman U, Vollgraf R, Shirvany R (2020) Proceedings of the 2020 SIAM international conference on data mining. SIAM, pp 55–63
15. Hajjar K, Lasserre J, Zhao A, Shirvany R (2021) Recommender systems in fashion and retail. Springer, pp 77–98
16. Dogani K, Tomassetti M, Vargas S, Chamberlain BP, De Cnudde S (2019) eCOM@ SIGIR
17. Caruana R (1997) Machine learning, vol 28, p 41
18. Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y (2013) arXiv:1312.6229
19. Eigen D, Fergus R (2015) Proceedings of the IEEE international conference on computer vision, pp 2650–2658
20. Kokkinos I (2017) Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6129–6138
21. Heuer F, Mantowsky S, Bukhari S, Schneider G (2021) Proceedings of the IEEE/CVF international conference on computer vision, pp 997–1005
22. Crawshaw M (200) arXiv:2009.09796
23. Zhang Z, Luo P, Loy CC, Tang X (2014) Computer vision–ECCV 2014: 13th European conference, Zurich, Switzerland, September 6–12, 2014, proceedings, Part VI 13. Springer, pp 94–108
24. Dai J, He K, Sun J (2016) Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3150–3158
25. Liu S, Johns E, Davison AJ (2019) Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1871–1880
26. Li D, Li X, Wang J, Li P (2020) Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 1553–1556

27. Misra I, Shrivastava A, Gupta A, Hebert M (2016) Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3994–4003
28. Gao Y, Ma J, Zhao M, Liu W, Yuille AL (2019) Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3205–3214
29. Raschka S (2018) arXiv:1811.12808

Innovations in Beauty, Personalization, and Advertising E-Commerce

Automated Material Properties Extraction For Enhanced Beauty Product Discovery and Makeup Virtual Try-on

Fatemeh Taheri Dezaki, Himanshu Arora, Rahul Suresh, Amin Banitalebi-Dehkordi, and Marjan Celikik

Abstract The multitude of makeup products available can make it challenging to find the ideal match for desired attributes. An intelligent approach for product discovery is required to enhance the makeup shopping experience to make it more convenient and satisfying. However, enabling accurate and efficient product discovery requires extracting detailed attributes like color and finish type. Our work introduces an automated pipeline that utilizes multiple customized machine learning models to extract essential material attributes from makeup product images. Our pipeline is versatile and capable of handling various makeup products. To showcase the efficacy of our pipeline, we conduct extensive experiments on eyeshadow products (both single and multi-shade ones), a challenging makeup product known for its diverse range of shapes, colors, and finish types. Furthermore, we demonstrate the applicability of our approach by successfully extending it to other makeup categories like lipstick and foundation, showcasing its adaptability and effectiveness across different beauty products. Additionally, we conduct ablation experiments to demonstrate the superiority of our machine learning pipeline over human labeling methods in terms of reliability. Our proposed method showcases its effectiveness in cross-category product discovery, specifically in recommending makeup products that perfectly match a specified outfit. Lastly, we also demonstrate the application of these material attributes in enabling virtual-try-on experiences which makes makeup shopping experience significantly more engaging.

F. T. Dezaki · H. Arora · R. Suresh · A. Banitalebi-Dehkordi · M. Celikik (✉)
BeautyTech, Amazon, Vancouver, Canada
e-mail: marjan.celikik@zalando.de

F. T. Dezaki
e-mail: fatimaat@amazon.com

H. Arora
e-mail: hiaror@amazon.com

R. Suresh
e-mail: surerahu@amazon.com

A. Banitalebi-Dehkordi
e-mail: aminbt@amazon.com

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_3

1 Introduction

The makeup industry has grown exponentially, with an abundance of products catering to various preferences and skin types. However, this extensive selection poses a challenge for consumers as they navigate through countless options to find the right products that align with their specific requirements. Furthermore, customers often face the frustration of not accurately perceiving the visual attributes of makeup products solely through online catalog images. This lack of certainty necessitates waiting for the product to arrive, leading to a less-than-ideal experience.

Traditionally, e-commerce websites have relied on various sources of information, such as brands, titles, product descriptions, reviews, and recommendations, to guide the purchasing decisions of customers. While these sources can enable ranking of products in a relevant manner but they often fall short in providing a comprehensive understanding of the underlying material properties of the makeup products. Material properties, such as color, finish type and glitter attributes, play a vital role for the customers while making purchasing decisions of makeup products. Customers often have a color in mind and tend to look for similar products within the same category or across categories. Many times they also are looking particular finish types or glitter shades which aren't available with typical catalog data. Moreover, customers with a dress seeking matching makeup face the challenge of finding cosmetics that complement their attire. Coordinating the makeup with the dress's color, style, and overall aesthetic is their priority to achieve a cohesive and polished look. Alongside that, Virtual-try on capability [1, 2] for makeup products helps makes customers shopping experience more captivating and accurate. Enabling virtual try on also requires extracting material properties from catalog images or textual information available.

To tackle these challenges, we propose an innovative approach that centers around extracting material properties from makeup products and leveraging this information to enhance product discovery and recommendations. We propose systematic pipeline to extract key material properties such as color, finish type and glitter color with the help of available product images. Our approach serves a dual purpose: augmenting product discovery capabilities and enabling a virtual try-on experience for makeup products. Using our approach, we can particularly improve recommendations using similar material attribute properties as well as match them to clothing attributes. By employing our approach, we can significantly improve recommendations by considering similar material attribute properties and matching them with clothing attributes. For instance, we can address the problem faced by customers who have a finalized dress for a specific occasion and require matching makeup suggestions based on color and glitter attributes. Our extraction pipeline enables precise and tailored product recommendations for such scenarios (can be seen in Fig. 1). It is important to note that these extracted material properties work in conjunction with the currently used attributes in makeup recommendation. This customization allows for a personalized and tailored recommendation experience based on individual preferences and requirements.

In our work, we propose a novel pipeline specifically designed for extracting makeup attributes. In makeup industry, eyeshadow products exhibit a huge diversity

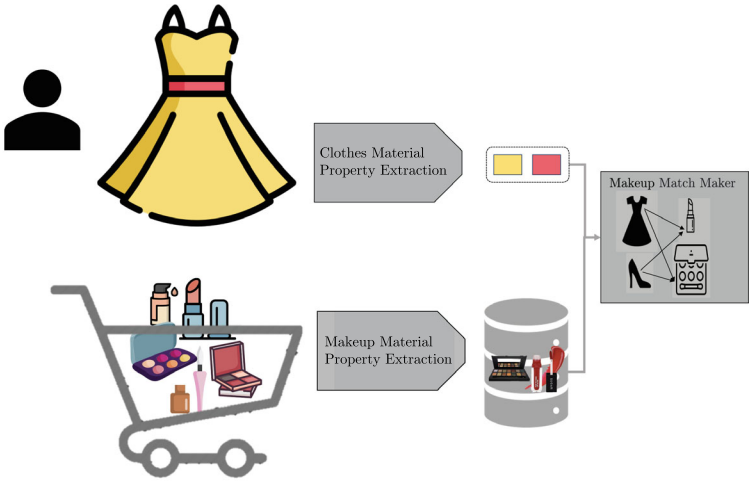


Fig. 1 Customer story and proposed solution: a customer has chosen a dress for a specific occasion and seeks matching makeup recommendations based on color and texture attributes

in terms of shade counts, base colors, and finish types, making them a challenging category to analyze. Therefore we consider the variety seen in eyeshadow products in the design of our pipeline, and then showcase the versatility and applicability of our approach to other makeup categories such as lipstick, foundations and etc. Our pipeline has been developed to effectively handle the broad range of shade counts typically found in eyeshadow products, spanning from as low as one to as high as 200. Moreover we design a pipeline for extraction of clothes material properties. Our integrated attribute extraction pipelines for makeup (eyeshadow, lipstick) and clothing enable cross-category product recommendations, enhancing the shopping experience. Our methodology delivers personalized suggestions across diverse makeup and fashion segments, catering to individual preferences and requirements. Our work is closely related to the topic of predicting object attributes and extracting attribute values from images. Prior research has primarily focused on attribute prediction in the fashion domain [3–5], such as extracting color, texture, and pattern from images of clothing items. In summary, our main contributions are:

- We design the material property extraction pipeline with the two-fold aim of enhancing product discovery experience and to facilitate creation of 3D assets required for virtual try-on applications.
- We propose an end-to-end learning pipeline for extracting all required material properties for makeup products.
- We demonstrate how our extracted attributes can improve relevancy of recommendations for makeup products both from makeup to makeup and outfit to makeup.
- We perform extensive experiments to show effectiveness and superiority of our learning pipeline and it's applicability for single and multi-shade eyeshadow products.

2 Method

In this section, we present an overview of our proposed pipeline (refer to Fig. 2). The pipeline comprises three main components: makeup material property extraction, clothes material property extraction, and makeup product match maker.

- Makeup Material Property Extraction: This component focuses on extracting material properties related to makeup products. It analyzes various characteristics such as format, finish type, color, and reflectivity. Firstly, we introduce the problem at hand for this main task and subsequently present our proposed solution in the form of an automated material property extraction pipeline. This pipeline aims to address the challenges faced in extracting key material attributes from makeup product images.

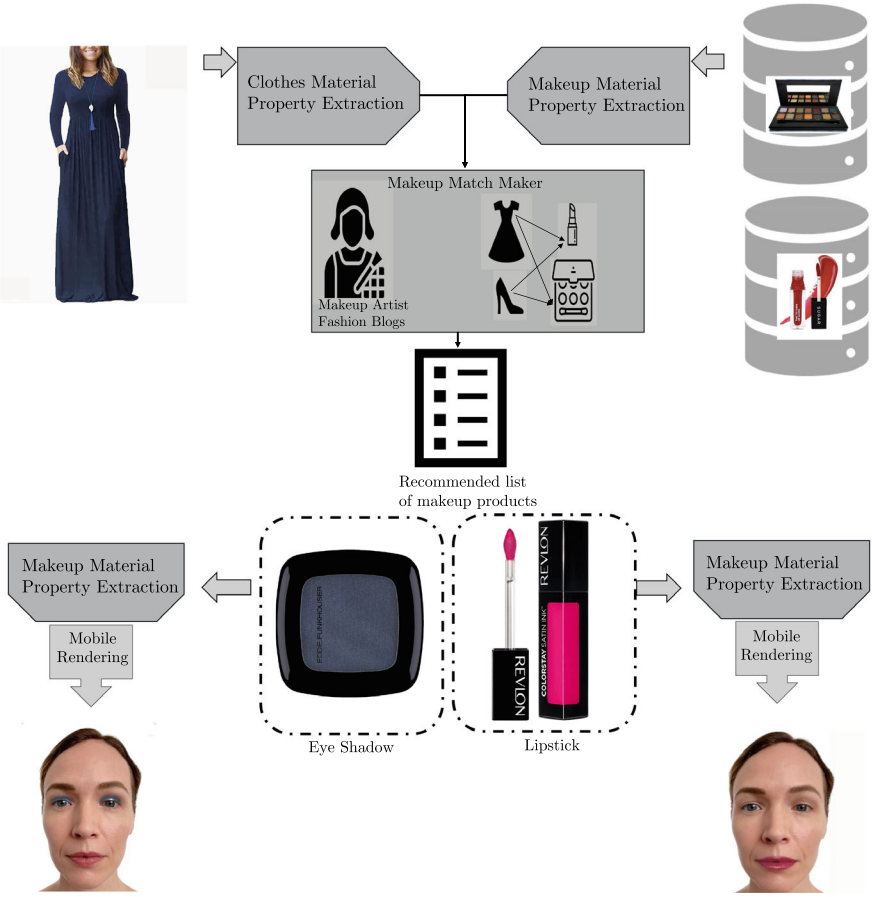


Fig. 2 Overview of the proposed pipeline: Three main components include makeup material property extraction, clothes material property extraction, and makeup product match maker

- Clothes Material Property Extraction: The second part of our pipeline is dedicated to extracting material properties specifically related to clothing items. By utilizing machine learning algorithms, we can effectively extract and categorize these properties from product images.
- Makeup Match Maker: The final component of our pipeline is responsible for finding the best matching makeup products based on the previous two components that align with the user's desired material properties. It leverages a comprehensive database of makeup products and their corresponding properties.

2.1 Makeup Material Property Extraction

An overview of our proposed eyeshadow MPE pipeline can be seen in Fig. 4. In this section, we first introduce the problem at hand and subsequently present our proposed solution in the form of an automated pipeline. This pipeline aims to address the challenges faced in extracting key material attributes from makeup product images, thereby improving the efficiency and accuracy of the process.

2.1.1 Problem Formulation

Online beauty retailers offer a wide range of eyeshadow products, and several factors contribute to the variety of these products available, including:

- **Brands**: Online beauty retailers offer a wide range of eyeshadow products from different brands, including popular drugstore brands like Maybelline and L'Oreal, as well as high-end brands like Chanel.
- **Colors and finishes**: Cosmetics come in a variety of colors/finishes, like matte, shimmer, metallic and glitter. Online beauty retailers offer a range of color options, from natural tones to bold and bright colors.
- **Formulations**: Eyeshadows are available in different formulations, such as powder, cream, stick and liquid. Each formulation offers a unique finish and texture, allowing customers to choose the one that best suits their needs.
- **Packaging**: Eyeshadow palettes are available in different packaging, such as individual pans, duos, quads, and larger palettes with multiple shades. This allows customers to choose the packaging that best suits their preferences and needs.

The key elements affecting the look of a beauty product applied on user's face includes application style, finish type, color, pigmentation/coverage, etc. From a user study, we identified the common application styles and concluded to have the material properties required for eyeshadow rendering as:

- **Format**: powder, cream, stick and liquid. Please see Fig. 3a for more illustration.
- **Base color** (RGB hex code). Please see Fig. 3b for more illustration.
- **Finish type** (one of): matte, shimmer, matallic, glitter. Please see Fig. 3c for more illustration.

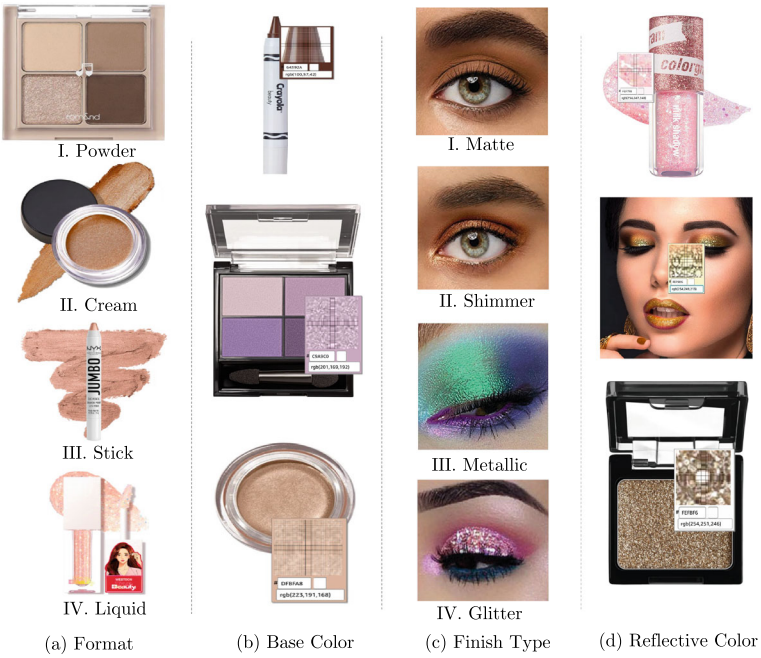


Fig. 3 Eyeshadow material properties

- **Reflective color** (if any)(rgb hex code). Please see Fig. 3d for more illustration.

In order to extract material properties of eyeshadow products from a detail page in an efficient and scalable manner, we propose a machine learning pipeline including several trained models specialized for each task in the MPE process.

2.1.2 Overall Proposed Model Architecture

Our MPE pipeline consists of six models each responsible for extracting a specific property of a given eyeshadow product. A visual description of the entire pipeline of eyeshadow MPE is shown in Fig. 4 and a detailed explanation of each component is provided in the following subsections. Given any eyeshadow product we should be able to extract the material properties required for eyeshadow rendering. The input to this pipeline is the product images and text descriptions of that eyeshadow product. At first, we have a simple filtering mechanism to check if the given product can be fed to the pipeline, for example some products are eyeshadow organizers/cases (they hold eyeshadow, but are not eyeshadow products) or the product shows makeup kits or bundles (there are more than just eyeshadow in the product) or they are multi-chrome eye or florescent/neon type eyeshadow which are not supported with the current rendering approach. For such filter we parse the product title and look for

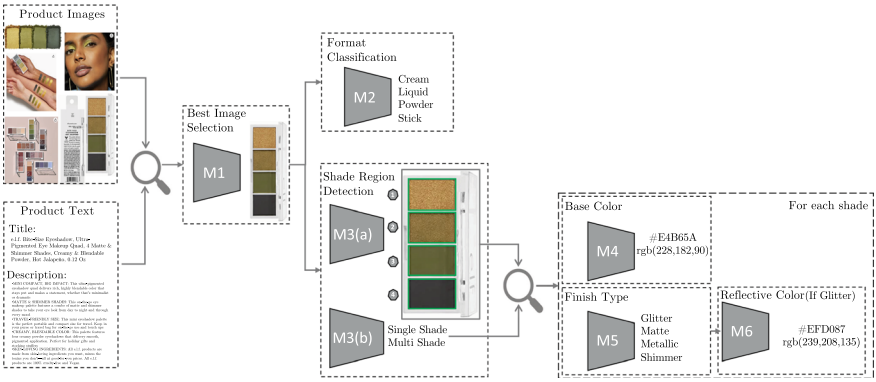


Fig. 4 Overall pipeline architecture

specific key words (and their variation), like *multichrome, iridescence, makeup kit, makeup organizer, ...* After this step, images are the only input for inferring the material properties of eyeshadow products.

To do so, as a next task we need to select the best representative image among, I^p_{best}, the varied images present for the product on the online beauty retailers' web pages, which is representative enough for MPE. The selected image is used as the input for the rest of the pipeline. First, it passes through a classification model to identify the format of the eyeshadow product, followed by a shade region detection model that detects the number of shades in the given product p and their locations in the picture. To ensure accuracy, we trained a binary classification model to predict whether the product p includes a single shade or more than one, which helps filter out incorrectly predicted regions from the previous model. For each detected shade, the base color and finish type are predicted using the region detected in the previous step. Finally, if the detected finish type is *glitter*, the reflective color is identified for rendering the sparkles using the last model in the pipeline.

2.1.3 Best Image Selection

In the online beauty retailers webpages, the catalog images have some orders to be displayed, but most often the orders don't imply any semantic information and also most sellers don't follow the prescribed order of image types. To fulfill our task, we require an image that displays all shades clearly, without being obstructed by any text or packaging. We perform this task by training a model which learns to choose the preferred image in a binary manner i.e. selects the 'preferred' image in a pair. We individually input two images into ResNet50 [6] based shared encoder, and then merge their features together to make a prediction about the preferred image. In the inference process, the trained model compares other images to the MAIN i.e. first image in the online beauty retailers' webpages image as a reference. If the model predicts any image as being preferred over the MAIN image, we select it for

downstream tasks. In the case where multiple images are predicted as preferred over the MAIN image, we choose the one with the highest confidence score to be used for the downstream tasks.

2.1.4 Shade Region Detection

Given the best image selected, I^p_{best}, we would like to locate all the existing shades with a bounding box: center point coordinates, width and height, $(x_i, y - i, w_i, h_i),\ i \in \{1, ..., s\}$. To do so, a popular object detection model, YOLOV5 [7], was trained on in-house dataset. In Fig. 5 some sample eyeshadow products with the corresponding bounding boxes to show each shade location is shown. YOLOV5 utilizes a single-stage object detection approach, meaning it processes the entire image only once, which allows it to perform real-time object detection on high-resolution images. YOLOV5 is based on a deep convolutional neural network that consists of a backbone, neck, and head. The backbone is responsible for extracting features from the input image, while the neck and head perform object detection and classification tasks. The backbone of YOLOV5 uses a variant of the CSPNet (Cross-Stage Partial Network) architecture [8], which is designed to improve the learning capability of convolutional neural networks by facilitating cross-stage feature aggregation. This architecture allows YOLOV5 to process input images efficiently while preserving spatial resolution. The neck of YOLOV5 is composed of a series of convolutional layers that are responsible for fusing features from different levels of the network. This process is called feature pyramid fusion, and it allows YOLOV5 to detect objects of different sizes and scales. The head of YOLOV5 consists of a set of convolutional layers that predict bounding boxes and class probabilities for detected objects. The output of the head is a set of bounding boxes with confidence scores and class probabilities, which are then used to identify and localize objects in the input image. To ensure the accuracy of our model, we trained a binary classification algorithm for eyeshadow products with a single shade. This allows us to predict whether the product p contains one or more shades, which in turn helps to filter out incorrectly predicted regions from the YOLOV5 model. A Resnet50 model was fine-tuned for this binary classification task. For instance, if the image is classified as a single-shade product,

Fig. 5 Bounding boxes overlayed on different eyeshadow products with blue color

but the YOLOv5 model detects multiple regions, we can select the region with the highest confidence score and pass it on to the subsequent stages of the pipeline.

2.1.5 Format-Finishtype Classification

The pipeline includes two classification models: one for detecting the format of the eyeshadow product based on the best image, I_{best}, and another for detecting the finish type of each eyeshadow shade, s, based on cropped regions of the same image, I^s_{best}. Various model architectures were experimented with for each of the classification tasks, and the best architecture was selected based on factors such as data size and final accuracy.

Out of all the models that were tested for format classification, EfficientNet-B3 [9] exhibited the best performance. It is part of the EfficientNet family of models, which are designed to achieve state-of-the-art performance on a range of computer vision tasks while minimizing the number of parameters and computations required. EfficientNet-B3 has a relatively large number of layers, with a total of 76 convolutional layers, and it uses a combination of depthwise separable convolutions, squeeze-and-excitation blocks, and a compound scaling method to achieve high accuracy and efficiency. One of the key innovations of the EfficientNet architecture is the use of a compound scaling method, which scales the width, depth, and resolution of the model simultaneously. This allows the model to achieve high accuracy while using fewer parameters and computations than other models of comparable size.

Among the different models we experimented with, Resnet50 [6] with 4 classes performed the best in the finish type classification task. The input to this model is the cropped region detected in the previous step. As the finish type classification task is highly imbalanced with respect to the 4 classes, with a greater number of Matte and Shimmer finishes compared to Metallic and Glitter finishes, we employed various techniques to balance the classes, including data augmentation and oversampling of the classes with lower populations.

2.1.6 Basecolor-Reflective Color Regression

To detect the base and reflective colors (if any) of the shade region detected in the previous Sect. 2.1.4, three values for normalized RGB channels are predicted. Resnet50 was demonstrated to be a suitable choice as the backbone for these models. The base color of available eyeshadow products on in our dataset exhibits a high degree of variability, ranging from the darkest to the brightest values in each of the RGB channels. On the other hand, reflective colors mostly tend to have brighter values. Based on this observation, we utilized the raw RGB values for detecting the base color, whereas the reflective color is scaled after being predicted. For instance, if we take into account the predicted values for the R channel, we apply an additional scaling using the equation $aR + b$, where $a = 0.4$ and $b = 0.6$, to prevent the model from producing trivial outputs regardless of the input it receives.

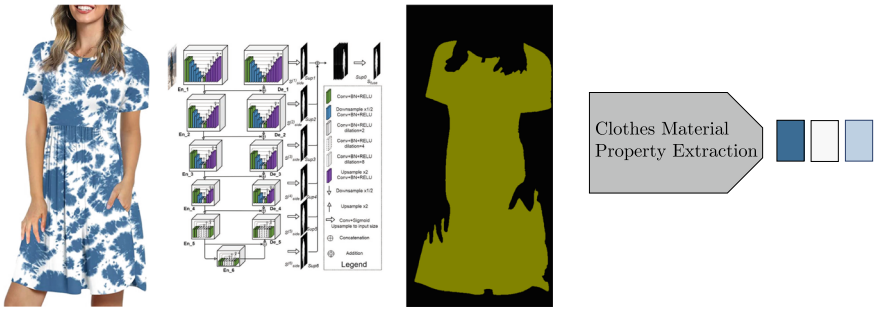


Fig. 6 Clothes color extraction from identified region using U2NET segmentation

2.2 Clothes Material Property Extraction

The clothes material property extraction process is comprised of two key components (refer to Fig. 6): a segmentation model and an attribute prediction model. For the clothes segmentation model, we employed U2NET [10], a deep learning model renowned for its effectiveness. Unlike the conventional salient object detection task where a single-channel output is obtained, our modified U2NET produces four channels. Each channel represents a specific clothing region, namely upper body cloth, lower body cloth, fully body cloth, and background. This model demonstrates excellent performance across various backgrounds and poses.

By leveraging U2NET's multi-channel output and employing appropriate loss functions, we could precisely identify and classify different clothing regions in an image. This allows for more precise analysis and prediction of clothing attributes in subsequent stages of the pipeline.

Once the desired region is identified, we proceed to extract the top three to five colors present in the clothes. To accomplish this, we employ the k-means clustering algorithm, which enables us to identify the dominant colors within the clothing region. These extracted colors play a crucial role in determining the best makeup product that complements the chosen clothing.

2.3 Makeup Match Maker

We leverage the extracted color and finish types to search for relevant products in our catalog, utilizing the extracted material attributes. We seek input from makeup professionals who suggest suitable makeup i.e.lipstick and eyeshadow options that complement the dresses. By incorporating material property extraction (MPE) attributes for makeup products, we strive to deliver recommendations that align with human preferences. Although we currently employ a manual matching strategy due to the subjective nature of preferences, automated rules can be established to scale the recommendation system.

3 Experiments

3.1 Datasets

Experiments were conducted on a collection of makeup/clothing product images crawled from online shopping webpages. The eyeshadow data utilized in the experiments comprised of 3700 products that appear to be top sold products on shopping websites. The eyeshadow products have varying numbers of images to display, with an average of 5 images per product. The number of images can range from a minimum of one to a maximum of twenty. The data was divided into two separate sets, a training set and a validation set, using a mutually exclusive split with a ratio of 70–30% respectively.

Figure 7 provides a comprehensive view of the eyeshadow dataset. The data shows that "Powder" is the most common format followed by stick, liquid, and cream, respectively. In addition, Fig. 7b reveals that most eyeshadow products have a single shade, although some have as many as 250 shades in one product. Matte is the most prevalent finish type, followed by Shimmer, with Glitter and Metallic being less common. Iridescent shades, which feature multiple colors in one shade, were filtered based on the text in detail page.

Furthermore, the 3D map of base color and reflective color in RGB space highlights the color distribution in Fig. 7d, e. While there are many variations in the base color of eyeshadow products, the reflective color is mostly concentrated in one corner of RGB space, with large values in all three channels of R, G, and B.

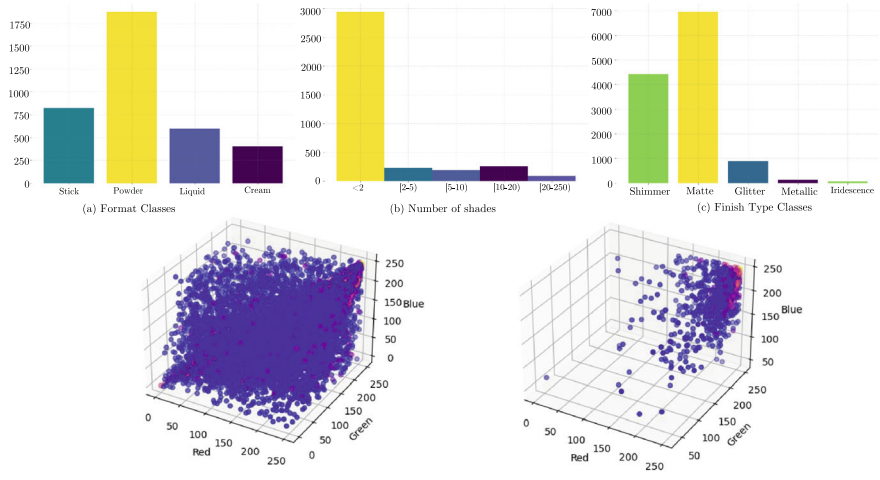


Fig. 7 Data visualization

3.2 Results

To evaluate the performance of each ML model in our pipeline we used different evaluation metrics and report them in Table 1. These metrics include

- **deltaE** [11] is a metric to quantify distance between two colors. It is computed as euclidean distance between two points in CIELAB colorspace.
- **mAP** [12] is mean average precision use to quantify how many objects are correctly localized in the prediction. We show mAP for objects with IoU between 0.5 to 0.95.
- **F1-Score** is computed by calculating harmonic mean of precision and recall of classification model.
- **Selection Accuracy** is the count of times ML model selection matches human selection.

Table 1 displays the performance of our models under multiple scenarios. In the first row, we use all the models' predictions to evaluate end-to-end impact. We also show separate analysis of single shade and multi shade for both finish type prediction(M5) and base color(M4) as they are the key end points. In the second row, we do not use Best Image Selection Model (M1) predictions and instead the GT data for image selection has been used. We observe that there's a slight difference which is propagated due to error in M1. For example M3 goes from 0.93 mAP to 0.94. Similarly, in following row we substitute both M1 and M3 (Shade Detection) with ground truth to observe error getting carry forward. In the last row, we use ground-truth for finish type prediction so we can observe the performance of M6. M6 delta E increases because more shades are Glitter in Ground Truth as compared to using M5 prediction model. We show that the deltaE for base color identification is 6 for single shade and 5 in multi-shade which is pretty good for color accuracy and falls within the range where it becomes difficult for an human to differentiate.

We demonstrate various types of eyeshadow products in Fig. 8 and compare the ground truth annotations with the predictions generated by our individual pipeline models. In the last column, we additionally compare the appearance of the rendered eyeshadow product on static images of a human model. As it can be seen our proposed

Table 1 End-to-end models performance for all individual tasks in the pipeline

				Single shade		Multi shade		
	M1 (Acc↑)	M2(f1-score↑)	M3(mAP↑)	M4(deltaE ↓)	M5(F1-Score↑)	M4(deltaE ↓)	M5(F1-Score↑)	M6(deltaE ↓)
	Best image selection	Format classification	Shade region detection	Base color Identification	Finish type classification	Base color identification	Finish type classification	Reflective color Identification
NO GT	92.22	0.90	0.93	6.35	0.84	4.86	0.89	6.18
GT M1	NA	0.91	0.94	6.35	0.84	4.76	0.89	6.18
GT M1 + M3	NA	0.91	NA	5.89	0.87	4.83	0.9	5.85
GT M1 + M3 + M5	NA	0.91	NA	5.89	NA	4.83	NA	6.28

Shade detection Info:
GT
Prediction
M3(a): map of 0.97
M3(b): Single shade (P= 0.99)

B092JPDYRR	Format	Finish type	Base color	Reflection Color	VTO
Ground truth	Powder	Shimmer	#DDCEC6	NA	
Prediction	1- Powder(p=0.99) 2- Liquid(p=2.3e-05) 2- Cream(p=2.05e-05) 4- stick(p=2.6e-08)	1- Shimmer(p=0.858) 2- Matte(p=0.095) 3- Glitter(p=0.044) 4- Metallic(p=0.002)	#D9C5BB (deltaE=3.77)	NA	

(a)

Shade detection Info:
GT
Prediction
M3(a): map of 0.99
M3(b): Single shade (P= 0.91)

B0BB1P5SMK	Format	Finish type	Base color	Reflection Color	VTO
Ground truth	Liquid	Glitter	#3C3839	#FDF8FD	
Prediction	1- Liquid (p=0.99) 2- Stick(p=0.01) 3-Cream (p=0.00) 4-Powder(p=0.00)	1- Glitter(p=0.69) 2- Shimmer(p=0.26) 3- Matte(p=0.03) 4- Metallic(p=0.00)	#54463a (deltaE=9.55)	#F5ECFB (deltaE=5.29)	

(b)

Shade detection Info:
GT
Prediction
M3(a): map of 0.98
M3(b): Multi shade (P= 0.89)

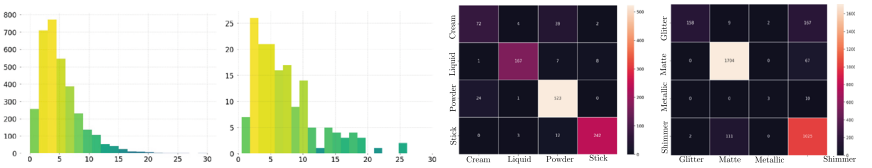
B07FXQVGX6	Format	Finish type	Base color	Reflection Color	VTO
Ground truth	Powder	Shimmer	#465254	NA	
		Matte	#343432	NA	
Prediction	1- Liquid (p=0.00) 2- Cream (p=0.01) 3- Powder(p=0.99) 4- stick(p=0.00)	1- Shimmer(p=0.65) 2- Matte(p=0.31) 3- Metallic(p=0.05) 4- Glitter(p=0.00)	#2d4844 (deltaE=7.48)	NA	
		1- Matte(p=0.77) 2- Shimmer(p=0.21) 3- Glitter(p=0.01) 4- Metallic(p=0.01)	#272428 (deltaE=6.9)	NA	

(c)

Fig. 8 End to end pipeline predictions and VTO on static model images

pipeline outputs are comparable to or even superior to that of human annotations. It can be further observed that despite minor differences with respect to Ground Truth in intermediate models, the final virtual try-on experiences are very close to as with ground-truth data. This demonstrates the robustness of our pipeline. Additional visualizations with higher resolution can be found in the supplementary material section.

In Fig. 9a, b, we present histograms of deltaE scores between the ground truth and our color models' predictions. The data shows that the majority of the deltaE



(a) Base color DeltaE histogram (b) Reflective color DeltaE histogram (c) Format classification confusion matrix (d) Finish type classification confusion matrix

Fig. 9 Performance visualization of DeltaE distance histograms of the base color and reflective color, and confusion matrices for format and finish type classification models

(a) Annotator A vs. annotator B (b) Annotator B vs. annotator C (c) Annotator A vs. annotator C

Fig. 10 Annotators agreement for finish type classification in our dataset

scores for both the base color and reflection color models fall below 10, which is an acceptable range. More specifically, our analysis shows that 30% of the base color model predictions have a deltaE score of less than or equal to 3, which is imperceptible to most human eyes, while 65% have a deltaE score within 3–12, which is distinguishable but still acceptable to human eyes.

Figure 9c, d shows the confusion matrix for our finish type and format classifications. While the finish type model performs well for most classes, it struggles with differentiating between *glitter* and *shimmer* finishes. We confirmed that this confusion is difficult for human annotators as well by studying their confusion for finish type classification labels in Fig. 10. Despite this confusion, our qualitative results demonstrate that these two classes are perceptually similar, and the final VTO rendering is nearly identical if the base color is predicted accurately.

In one of our experiments we evaluated the performance of base color and finish type ML models with respect to product images from different makeup brands. The experiment showed that both models performed better than average for key brands with top eyeshadow products. This is due to more uniform image quality from these larger brands. The experiment also helped identify brands with unsuitable images for MPE tasks, such as Revlon, showing a lower finish type F1-score than the average. More detailed results of this experiment can be seen in Supplementary Material section.

3.3 Ablation Studies

To assess the inter-human consistency, we gathered a group of individuals to perform the same task of annotating eyeshadow material properties. We then calculated the level of agreement between the three annotators for two main eyeshadow properties: base color and finish type. By comparing the performance of the ML model with the inter-annotator agreement among the three annotators, we can gauge the level of agreement between the annotators and determine if the model is performing better or worse than humans on average.

As for assessing base color model predictions and measuring the consistency of our model and that of human annotators, we compute the discrepancy among human

Table 2 Human consistency analysis

	Single-shade		Multi-shade		All-shades	
	Mean	Variance	Mean	Variance	Mean	Variance
d_{HC}	6.31	62.48	11.82	112.84	6.42	64.10
d_{ML}	6.36	34.43	10.2	59.95	6.44	35.24

Table 3 Fleiss kappa coefficient and F1-score for finish type and format classification

	Fleiss-Kappa	f1-score
Format prediction	0.91	0.902
Finish type prediction	0.66	0.870

annotators for any shade s as:

$$d_{HC} = dist(A1_s, A2_s) + dist(A1_s, A3_s) + dist(A2_s, A3_s)$$

where $dist(x, y)$ refers to the deltaE distance between two colors and $A1_s$, $A2_s$, $A3_s$ refers to color annotated by each of three annotators for shade s. Similarly, we compute the discrepancy for each shade our ML model using

$$d_{ML} = dist(A1_s, ML_s) + dist(A2_s, ML_s) + dist(A3_s, ML_s)$$

where ML_s refers to predicted base color by ML model for shade s. We show the mean and variance in the Table 2. Assuming the mean of human annotations as ground-truth, we observed that the ML model's distance to the ground-truth is comparable to the sum of differences among human annotations. However, the variance in the ML model's predictions is significantly lower, indicating that the model is less likely to deviate from the ground-truth. This suggests that the ML model provides consistent and less subjective ratings compared to human annotators.

As for analyzing performance of our classification tasks, we also compare the F1-score of ML model predictions with the Fleiss Kappa coefficient computed on human annotations. Fleiss Kappa coefficient is a commonly used measure of inter-rater agreement for categorical items [13, 14]. As can be seen in Table 3, we observe that our f1-score is similar to the Fleiss Kappa coefficient for the format classification task, and significantly better for finish type prediction. This indicates that our ML model is equally or more reliable than human annotations for both classification tasks.

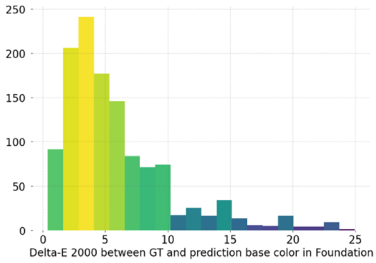
3.4 Extension to Other Makeup Categories

The versatility of our proposed method extends beyond eyeshadow alone, as it can be effectively applied to other makeup categories such as lipstick, foundation, and more. To demonstrate its adaptability, we evaluated the performance of our

pre-trained model by testing it on datasets from these additional categories, which further validates its broad applicability.

For this experiment, a total of 1,240 "foundation" product pages were crawled from online shopping websites. Each of these products was meticulously annotated for color information, ensuring a comprehensive dataset for our analysis. The model demonstrated promising performance, with an average deltaE score of 5.86. In Fig. 11a, we present histograms illustrating the distribution of deltaE scores between the ground truth and our color models' predictions. The data analysis reveals that the majority of the deltaE scores for the base color models are below 10. Figure 11b showcases two representative samples of foundation products alongside their corresponding model predictions and ground truth.

In another experiment, we gathered a dataset of 2,165 "lipstick" products. This dataset was carefully annotated, and the experiment was conducted following the same procedure as mentioned earlier. The average deltaE score obtained for this experiment was 7.91. The distribution of deltaE scores is visualized in Fig. 12a, and



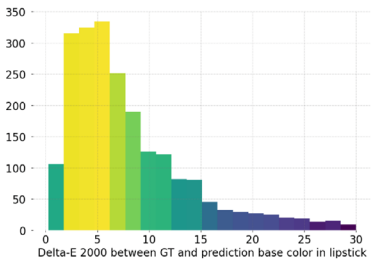
(a)

B006WRDX72	Format	Finish type	Base color	
Ground truth	NA	NA	#C59D7C	
Prediction	1- Liquid (p=0.96) 2- Cream(p=0.03) 3- Powder(p=0.015) 4- Stick (p=0.005)	1- Matte (p=0.999) 2- Glitter(p=0.0) 3- Metallic(p=0.00) 4- Shimmer(p=0.00)	#B08C6E (deltaE=6.03)	

B01MYF2RB7	Format	Finish type	Base color	
Ground truth	NA	NA	#B39180	
Prediction	1- Powder (p=0.93) 2- Cream(p=0.04) 3- Liquid (p=0.02) 4- Stick (p=0.01)	1- Matte (p=0.90) 2- Shimmer (p=0.09) 3- Glitter (p=0.00) 4- Metallic (p=0.00)	#B29889 (deltaE=4.49)	

(b)

Fig. 11 Performance visualization of DeltaE distance histograms for the base color predictions in foundation makeup products along with two sample results



(a)

B000H3DC4Q	Format	Finish type	Base color	
Ground truth	NA	NA	#B89188	
Prediction	1- Stick (p=0.86) 2- Liquid (p=0.12) 3- Cream (p=0.01) 4- Powder (p=0.005)	1- Shimmer(p=0.81) 2- Matte(p=0.18) 3- Glitter(p=0.05) 4- Metallic(p=0.05)	#e16a4f (deltaE=13.02)	

B00M25WE8W	Format	Finish type	Base color	
Ground truth	NA	NA	#af6254	
Prediction	1- Liquid (p=0.98) 2- Stick (p=0.01) 3- Cream (p=0.006) 4- Powder (p=0.004)	1- Matte (p=0.91) 2- Shimmer (p=0.085) 3- Glitter (p=0.004) 4- Metallic (p=0.001)	#A66A5F (deltaE=3.75)	

(b)

Fig. 12 Performance visualization of DeltaE distance histograms for the base color predictions in lipstick makeup products along with two sample results

Fig. 13 Recommendation makeup products for various clothing dresses. The text on each item is the makeup recommendation for each of the dresses by our in-house makeup professional

sample results are presented in Fig. 12b. Notably, a deltaE score below 5 indicates that two colors are perceived as highly similar by human observers.

3.5 Impact in Product Recommendations

Figure 13 showcases the effectiveness of our proposed Makeup Match Maker in providing complementary makeup recommendations for clothing items. The Makeup Match Maker demonstrates its versatility by successfully accommodating various clothing varieties and different types of eyeshadow and lipstick products, all based on the extracted material properties. In Fig. 14, we demonstrate the application of similar material properties for eyeshadow products, enabling intra-product recommendations. For our experiments, we set a deltaE color distance limit of 10 and rank the closest color products to demonstrate the effectiveness of our approach.

Additionally, Fig. 15 provides a qualitative demonstration of how the incorporation of our material attributes influences the recommendations. The baseline method primarily relies on brand and product form for recommendations. However, our approach significantly enhances the relevance of the recommendations, as observed in the displayed results.

In real-world applications, our approach can be combined with other structured attributes and ranked based on user shopping history. Figure 16 demonstrates how we incorporate extracted properties with available attributes like brand or product type to enhance recommendations. For instance, if a user's initial search query mentions a stick eyeshadow and a specific color preference, our recommendation system

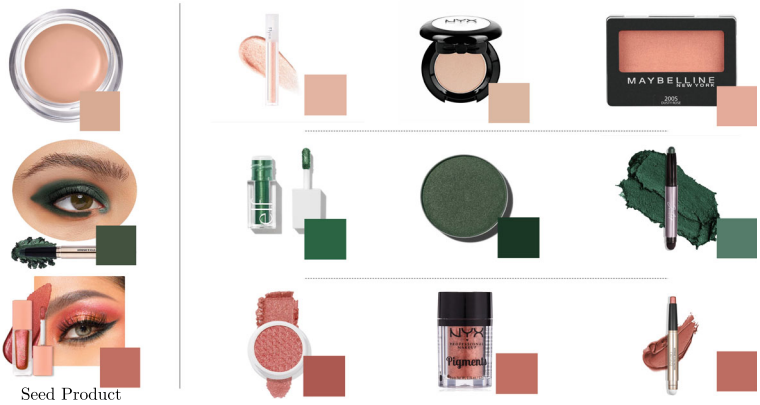


Fig. 14 Recommended eyeshadow products on basis of color similarity

Fig. 15 Recommendations for makeup products. **a** shows the baseline recommendations which don't include our material attributes and **b** shows recommendations which have material attributes incorporated

intelligently combines these attributes to suggest products that encompass both criteria. This exemplifies our ultimate objective of leveraging these crucial properties to impact recommendations and align them more closely with users' intent when searching for makeup products. To evaluate the effectiveness of our recommendation algorithm, we conducted an internal user study with 60 participants from diverse backgrounds. Each participant was presented with a combination of recommended products generated using both the baseline and our updated strategy. The task assigned to the participants was to select the item they were most likely to click on, based on the current item. During the study, we evaluated 20 products for both lipstick and eyeshadow with each participant. The results demonstrated that 78% of the participants chose to click on products recommended by our strategy. This outcome from

Fig. 16 Recommendations for eyeshadow products depending on search query by customer. On the left we have the product and on the right we show the recommended products given the initial search query

the user study provides robust evidence for the significance of material attributes in makeup recommendations. It reinforces the notion that incorporating these attributes substantially improves the relevance and appeal of our recommendations to users.

4 Conclusion

We proposed a novel learning-based pipeline that utilizes multiple neural network models to extract material properties from makeup products. Our comprehensive pipeline encompasses various models dedicated to image selection, format classification, shade region detection, base/reflective color estimation, and finish type classification. This approach allows for accurate material property extraction from eyeshadow products. Furthermore, we demonstrated the scalability of our model to other makeup categories, such as lipstick and foundation products. Additionally, we showcased the efficacy of utilizing these extracted material properties for successful cross-category product matching.

References

1. de Fátima Soares A, Morimoto BCH (2019) 21st symposium on virtual and augmented reality (SVR). A virtual makeup augmented reality system 2019:34–42. https://doi.org/10.1109/SVR.2019.00022
2. Kips R, Jiang R, Ba S, Duke B, Perrot M, Gori P, Bloch I (2022) Real-time virtual-try-on from a single example image through deep inverse graphics and learned differentiable renderers. Comput Graph Forum 41(2):29–40

3. Gutierrez P, Sondag P-A, Butkovic P, Lacy M, Berges J, Bertrand F, Knudson A (2018) Deep learning for automated tagging of fashion images. In: Proceedings of the European conference on computer vision (ECCV) workshops
4. Guo S, Huang W, Zhang X, Srikhanta P, Cui Y, Li Y, Adam H, Scott MR, Belongie S (2019) The imaterialist fashion attribute dataset. In: Proceedings of the IEEE/CVF international conference on computer vision workshops
5. Adhikari SS, Singh S, Rajagopal A, Rajan A (2019) arXiv:1907.00157
6. He K, Zhang X, Ren S, Sun J (2016)Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770-778
7. Jocher G, Chaurasia A et al (2022) ultralytics/yolov5: v7.0 - YOLOv5 SOTA realtime instance segmentation. https://doi.org/10.5281/zenodo.7347926
8. Wang C-Y, Mark Liao H-Y, Wu Y-H, Chen P-Y, Hsieh J-W, Yeh I-H (2020) Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, CSPNet: a new backbone that can enhance learning capability of CNN, pp 390–391
9. Tan M, Le Q (2019) Efficientnet: rethinking model scaling for convolutional neural networks. In: International conference on machine learning, pp 6105–6114
10. Qin X, Zhang Z, Huang C, Dehghan M, Zaiane OR, Jagersand M (2020) Pattern Recognit 106. https://www.sciencedirect.com/science/article/pii/S0031320320302077, https://doi.org/10.1016/j.patcog.2020.107404
11. Ortiz-Jaramillo B, Kumcu A, Philips W (2016) 2016 Eighth international conference on quality of multimedia experience (QoMEX), evaluating color difference measures in images, pp 1–6
12. Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2009) IEEE Trans Pattern Anal Mach Intell 32(9)
13. He J, Huang X, Gao H, Zhang Y, Wang Y (2021) Pattern Recognit 111
14. Fleiss JL (1971) Measuring nominal scale agreement among many raters. Psychol Bull 76(5):378
15. Liu J, Lu H (2018) Proceedings of the European conference on computer vision (ECCV) workshops, deep fashion analysis with feature map upsampling and landmark-driven attention
16. Kips R, Bokaris P-A, Perrot M, Gori P, Bloch I (2022) arXiv:2202.03723
17. Statista Online Community. Consumer Market Insights, Statista (2023). https://www.statista.com/outlook/cmo/beauty-personal-care/worldwide
18. Li T, Qian R, Dong C, Liu S, Yan Q, Zhu W, Lin L (2018) Proceedings of the 26th ACM international conference on multimedia, BeautyGAN
19. Nguyen T, Tran AT, Hoai M (2021) Lipstick ain't enough: beyond color matching for in-the-wild makeup transfer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 13305–13314

UNICON: A Unified Framework for Behavior-Based Consumer Segmentation in E-Commerce

Manuel Dibak, Vladimir Vlasov, Nour Karessli, Darya Dedik, Egor Malykh, Jacek Wasilewski, Ton Torres, and Ana Peleteiro Ramallo

Abstract Data-driven personalization is a key practice in fashion e-commerce, improving the way businesses serve their consumers' needs with more relevant content. While hyper-personalization offers highly targeted experiences to each consumer, it requires a significant amount of private data to create an individualized journey. To alleviate this, group-based personalization provides a moderate level of personalization built on broader common preferences of a consumer segment, while still being able to personalize the results. We introduce UNICON, a unified deep learning consumer segmentation framework that leverages rich consumer behavior data to learn long-term latent representations and utilizes them to extract two pivotal types of segmentation catering various personalization use-cases: *lookalike*, expanding a predefined target seed segment with consumers of similar behavior, and *data-driven*, revealing non-obvious consumer segments with similar affinities. We demonstrate through extensive experimentation our framework's effectiveness in fashion to identify lookalike Designer audience and data-driven style segments. Furthermore, we present experiments that showcase how segment information can be incorporated in a hybrid recommender system combining hyper and group-based personalization to exploit the advantages of both alternatives and provide improvements on consumer experience.

Manuel Dibak, Vladimir Vlasov, Nour Karessli Authors contributed equally to this work.

The original version of the chapter has been revised. A correction to this chapter can be found at https://doi.org/10.1007/978-3-031-76878-1_8

M. Dibak · V. Vlasov · N. Karessli · D. Dedik · E. Malykh · J. Wasilewski · T. Torres · A. P. Ramallo (✉)
Zalando SE, Berlin, Germany
e-mail: manuel.dibak@zalando.de

V. Vlasov
e-mail: vladimir.vlasov@zalando.de

N. Karessli
e-mail: nour.karessli@zalando.de

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_4

1 Introduction

Personalizing experiences such as search, ranking and recommendation in e-commerce is a fundamental capability to provide consumers with relevant content that is tailored to their specific preferences. Group-based personalization [1–7], in comparison to hyper-personalization [8–11], allows serving consumers with a customized experience based on broader patterns observed in the grouped segments. This can be particularly beneficial in addressing the cold start problem [2] appearing in new consumers with limited amounts of signals as well as low-engaged consumers whose historical signals can be outdated. Profiting from collective preferences within a segment, group-based personalization improves the quality of recommendations when individual recommendations are not relevant [12] as well as the diversity of the recommended content encouraging consumers to explore new and more diverse items [13]. The quality of group-based personalization heavily depends on the quality of the segments and their representation of the user base. The majority of the reviewed work on group-based personalization assumes consumer segments already exist, based on external social interactions like viewing the same TV program or touristic packages [14, 15], going to the same venue [6, 7] or having the same friends networks [16]. Another common segmentation criterion are ratings of the same movie [4, 5, 17].

In this work, we develop a method to identify high quality consumer groups that can be used to improve personalized experiences and propose *UNICON—a UNIfied framework for behavior-based CONsumer segmentation in e-commerce*. We distinguish between two important types of consumer segmentation: (1) Lookalike segmentation, where given a predefined seed audience segment (often defined by a business rule), the goal is to expand this segment using ML-based models that use consumer behavior data and predict those that don't (yet) match the business rule but are behaving highly similar to the seed segment [18, 19] (e.g., consumers that have never purchased a certain brand but may have affinity towards it), and (2) Data-driven segments, where the groups emerge from the consumer data and allows us to find non-obvious segments of similar consumers that are too complex to be captured with a set of rules [16, 20, 21].

Our work's main contributions can be summarized as follows: (1) a consumer unified long-term representation using rich consumer behavior data in a multi-headed self attention transformer-based [22, 23] model, (2) two methods that utilize the latent embeddings to extract two types of consumer segments, namely, lookalike and data-driven segments and apply on real-world data at scale to define lookalike Designer segment and data-driven style-based segments, and (3) experiments and results for real life use cases with both data segments, one incorporating it in a hybrid recommender system in a product carousel and the other to improve the experience for Designer consumers in the catalog ranking. Figure 1 illustrates the proposed framework.

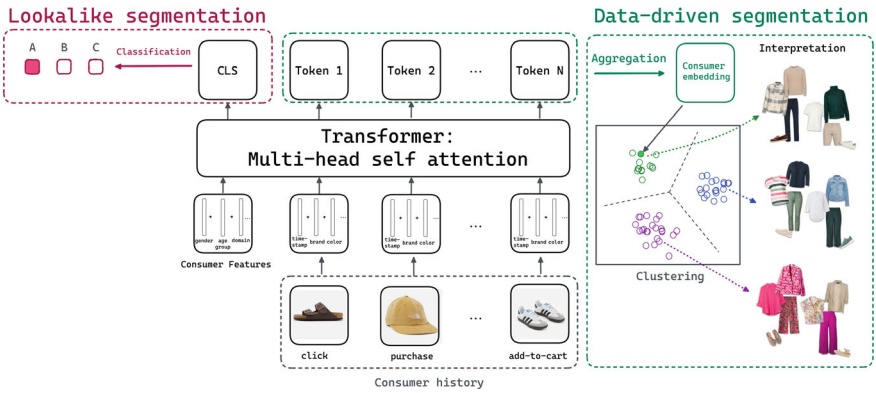


Fig. 1 Schematic figure of UNICON: A sequence of consumer interactions is embedded into tokens. A multi-head self attention transformer network generates embeddings for these tokens which are then used for lookalike and data-driven consumer segmentation

The remainder of this paper is structured as follows: in Sect. 2 we discuss the related literature; in Sect. 3 we detail the proposed UNICON framework and methods; in Sect. 4 we share offline and online experimental results; finally, in Sect. 5 we conclude the work and briefly discuss future work.

2 Related Work

2.1 Consumer Segmentation

The exploration of lookalike modeling and data-driven segmentation approaches for recommender systems remains an ongoing area of investigation in the scientific literature. In [20] authors perform consumer segmentation based on the properties of ordered shirts. In [16] authors propose a model that highlights similar users based on their behavior and friend network using neural networks. The users who appear in friend circles are used to build the group profile. Personalized search results are created by combining individual and group profiles with respect to the current query. [21] introduces a real-time attention based lookalike model that dynamically captures user preferences and behavior in real-time to identify similar users. By incorporating attention mechanisms, the model can effectively weigh the importance of different user attributes and adapt to changing user preferences. The model is trained using true similarity scores known for the training dataset. In a related line, [19] presents an innovative method for cross service lookalike modeling to improve user-targeting in campaigns. Authors propose a rule-based associative classification model that identifies meaningful associations between user attributes and conversion likelihood. By leveraging these associations, the model is able to identify users with similar characteristics, leading to enhanced conversion rates. More specifically in the context

of fashion, segmentation based on style has been explored with the help of computer vision methods. In [24] authors detect local periodically recurring fashion trends from images taken at the same city. In [25] images from photo-sharing services and social media platforms were analyzed using computer vision models to find global and per-city fashion choices and spatio-temporal trends. In [26] authors designed a "game" to get training data by letting a user compare two images related to their style, e.g. "Who's more hipster?" and use the collected data to build an inter style classifier ("Is this image goth or hipster?") and intra style ranker ("How hipster is this image?").

2.2 Clustering Approaches

In the data-driven segmentation case, no additional information about consumers such as lookalike labels, friend network or true similarity scores is known. Therefore, we leverage unsupervised clustering approaches. Our consumers are represented as a sequence of interactions with clothing articles. The problem of grouping sequences of tokens is reminiscent of the topic modeling problem in NLP. In [27, 28] authors propose to use document encoders [29] or sentence sequential encoders [30] as embeddings then use clustering algorithms. The main idea of these methods is to use Large Language Models (LLMs) to learn dense representation of documents or sentences in an embedding space. Since similar sentences and documents have similar dense representations, the embedding space can be clustered using unsupervised clustering such as k-means [31] or HDBSCAN [32]. The clusters can be labeled by collecting the most common words from the documents or sentences in the cluster. Other works [33, 34] propose to learn embeddings and groups jointly using deep clustering methods. The evaluation of such topic models is performed by sampling the documents or sentences from a cluster and asking a human or another LLM if these documents belong to the same topic. In this work, we use separate embedding and clustering approaches. To this end, we leverage an already existing recommendation model [22], which we adapt to generate consumer embeddings, and utilize clustering approaches to identify coherent groups. We ask fashion experts to evaluate if consumers in the same clusters have similar fashion styles.

2.3 Group Recommendations

Group recommendation systems [2, 3, 5] are mostly related with the problem of generating recommendations to predefined groups of individuals. They distinguish between persistent and ephemeral groups. While the first one can be considered static groups [1, 4], like families, the latter are considered groups that spontaneously form for a specific activity [6, 7], e.g., a group of friends coming together to go to a restaurant. In these works attention mechanisms are used to capture user preferences and model group interactions.

Unlike the previous methods, in this work we provide a unified framework for identifying consumer segments in either supervised or unsupervised manner and leveraging these segments in a hybrid recommender system.

3 Methodology

We represent consumers as a sequence of interaction (article-click, add-to-cart, add-to-wishlist and checkout) on articles identified by their stock keeping unit (SKU). Using a transformer-based model, we can learn the consumer behavior and subsequently find segments of consumers that interact similarly.

We identify two separate ways of defining segments, *Data-driven segmentation* and *Lookalike segmentation*. Both methods can be summarized in the *UNICON* framework, which involves the steps below. Figure 1 illustrates an overview how both types of segmentation arise from the consumer histories.

1. Data preparation: This involves selecting the right time frame of interactions and the right attributes of the interacted items to naturally form segments of certain properties. For lookalike segmentation this also involves identifying consumers that fulfill the business logic and can be used as training examples.
2. Next item prediction training (data-driven only): This trains the transformer to predict the next item (token) the consumer will interact with by using a softmax classifier with cross entropy loss and masking out future interactions. This ensures the model finds an internal representation of the consumer and naturally groups sequences together that would likely interact with the same items and therefore are similar in behavior.
3. Classification training (lookalike only): This step trains the model to predict membership of the consumer segment based on the business logic. The classification uses the CLS token as input and produces scores for each of the classes through a softmax layer.
4. Embedding extraction (data-driven only): This aggregates the embeddings of all the interactions of the consumer into a single point in a high-dimensional space in which proximity corresponds to similarity in terms of behavior, meaning that consumers close together will likely interact with the same type of items.
5. Clustering (data-driven only): In a final step, the groups are found using clustering to identify segments in the embedding space, which correspond to segments of consumers.

3.1 Data-Driven Segmentation

Data-driven segmentation aims at learning to partition consumers into segments solely based on their interactions. This approach uses unsupervised learning to group consumers together that exhibit similar behavior patterns. For data driven segmentation, we deploy a two-step approach: (1) we learn consumer embeddings in which

consumers exhibiting similar behavior are close to each other by a distance, or similarity metric. Subsequently, (2) we divide this space into groups such that a consumer can belong to only one group.

Consumer Embeddings. We employ a transformer-based model to compute consumer embeddings from a sequence of interactions with items. Each item in the input sequence is represented by various attributes such as brand, color, silhouette, etc. The model is trained to predict the SKU of the next item interaction using a causal self-attention mechanism and a categorical cross-entropy loss [22]. The consumer embeddings are extracted from the encoder's output, by averaging over the embedded token sequence. All interactions are treated equally in the averaging.

Segmentation. Once the embedding has been created, the segmentation is performed by applying a clustering algorithm on the consumer embeddings.

3.2 Lookalike Segmentation

Lookalike segmentation on the other hand use a set of consumers defined by business logic and aims at finding consumers that behave similarly to them, i.e., aims at extending groups of consumers defined by business rules. This approach learns the typical behavior of such segments based on their interaction sequences in order to find consumers that behave similarly and would likely fulfill the business rules in the future. Formally, let C be the set of consumers fulfilling the business logic. We call this set of consumers the *core segment*. Further, let C^E be the set of consumers that in the future will become part of the core segment. We are interested in identifying the set of *lookalike consumers* L, which will likely be part of the core segment in the future:

$$L = \left\{ l \mid l \notin C \wedge p\left(l \in C^E\right) > \tau \right\}, \tag{1}$$

meaning that their probability of becoming a core consumer in the future is higher than some pre-defined threshold τ.

This definition of lookalike consumers allows for a formulation of the identification of the *lookalike segment* as a supervised learning problem, where we use the set of core consumers as positive examples, and random non-core consumers as negative examples for a binary classifier based on the user history.

4 Experiments and Results

We identified two use cases to apply data-driven and lookalike segmentation. For data-driven segmentation, we aim at identifying segments of consumers that show affinity to a similar fashion style. This is a challenging task, as there is no straightforward way of quantifying a style similarity. As a part of this project, we propose a method to address this problem and to automate the evaluation of the consumer

segments. As an application of lookalike segmentation, we identify consumers that are likely to be interested in designer items. Designer items are those produced by prestigious and luxurious brands. To this end, we define *designer consumers* based on rules applied to consumers' histories and apply the method to find a set of *designer lookalikes*.

4.1 Data-Driven Style Segmentation

As we want to find segments of consumers that are similar in style, we need to only present the model with information that is relevant for style identification. We achieve this by filtering the consumer histories to only contain information about interaction that might indicate a certain style affinity of the consumer.

Data variants
We explore several data variants: **Baseline)** consumer sequences with interactions (article-click, add-to-wishlist, add-to-cart, and checkout) on any SKUs over the previous two months. These sequences can have mixed SKUs in terms of fashion preference (female, male) (39.1M sequences). **(V1)** filtering consumer sequences to contain only style relevant silhouettes defined by a fashion expert (37M sequences). **(V2)** V1 + splitting consumer sequences by fashion gender preference (26.4M male sequences and 33M female sequences). **(V3)** V1 + removing sequences that contain only a single silhouette type (28.7M sequences). **(V4)** Combine the filters of variants V1 and V3 (16.5M male sequences and 24.1M female sequences).

Consumer embedding and segmentation offline experiments
The goal is to validate that proximity in the embedding space reflects similarity in style. Additionally, this allows us to identify the optimal distance metric in the embedding space as the one that correlates most with style similarity.

As a first step, we define a style similarity score S between two users u_1 and u_2 as

$$S(u_1, u_2) = \sum_{a \in A} w_a \left[1 - \mathrm{JSd}\left(P(a, u_1) || P(a, u_2)\right)\right], \tag{2}$$

where A is the set of item attributes (e.g. brand, commodity group, color, etc.). w_a is the weight assigned to the attribute a, which is selected to match the behavior on samples of user histories annotated by a fashion expert. $P(a, u)$ is the distribution of item attribute a values in the consumer interaction history u. $\mathrm{JSd}(P||Q)$ is the Jensen-Shannon divergence. We use this score to evaluate the embedding space variants and to determine which distance metric to use for clustering. To this end, we compute the Pearson correlation between style similarity and distance metrics in the embedding space based on randomly sampled consumer pairs. Table 1 reports Pearson correlation of style similarity score, with three distance metrics Dot product, Cosine Similarity and Euclidean distance. We can see that all distance metrics show moderate correlation with the style similarity and representation of style increases

Table 1 Evaluation of the embedding space: Pearson correlation between style similarity defined by Eq. 2 and different distance/similarity metrics in the consumer embedding space

Distance/similarity metric	Baseline	V1	V2	V3	V4
Dot product ↑	0.435	0.471	0.481	0.476	**0.535**
Cosine similarity ↑	0.451	0.482	0.496	0.494	**0.545**
Euclidean distance ↓	−0.398	−0.409	−0.351	**−0.456**	−0.433

Table 2 Evaluation of the number of segments using k-means clustering

Number of segments	20	250	1000
Silhouette score (cosine) ↑	0.096	**0.097**	0.087
ROC-AUC style similarity ↑	0.77	0.88	**0.91**

when more restrictions are applied to data. We choose Cosine similarity and V4 as they exhibit the highest correlation with style similarity score.

To measure if the segments are well separated in the embedding space we use Silhouette Score [35] which measures how similar data points are to their own segment as compared to other segments. The score ranges between -1 and 1, with a higher score indicating a better cluster separation. Furthermore, to analyze whether the segmentation is reasonable in terms of style similarity, we use the area under the receiver operating characteristics curve (ROC-AUC) as a consistency metric between style similarity and the consumer segmentation. We compute ROC-AUC for pairs of consumers where style similarity is used as a score and the label is whether they belong to the same style segment.

Table 2 shows a comparison between methods with different numbers of segments. It can be seen that ROC-AUC is better for a large number of segments while the silhouette score is marginally better for a lower number. To find the right number of segments, we further investigate the distribution of distances to the segment center for points inside the segments. To this end, we compute pairwise distances between random pairs of consumer embeddings in the embeddings space. We then bin the consumer pairs based on this distance. Additionally, we compute the style-similarity between these pairs. We then average the style similarity in each of the bins. The resulting averaged style similarity as a function of the pairwise embedding distance is shown in Fig. 2 (top). We observe an approximate exponential decay of the form $\propto \exp(-d/\lambda)$. This allows for a definition of a correlation length λ, which determines the length in the embedding space at which the style similarity decays. We extract this length from an exponential fit to find a length-scale of $\lambda = 0.98$. This serves as an approximate upper bound to the cluster diameter, as clusters larger than λ would contain consumer pairs, which are no longer similar in style.

Figure 2 (bottom) shows the average and standard deviation of center distances per segment. The red line indicates the style similarity length scale λ as defined above, which approximates the distance in the embedding space at which the style similarity considerably decays. We chose 250 segments because it is a good trade-off

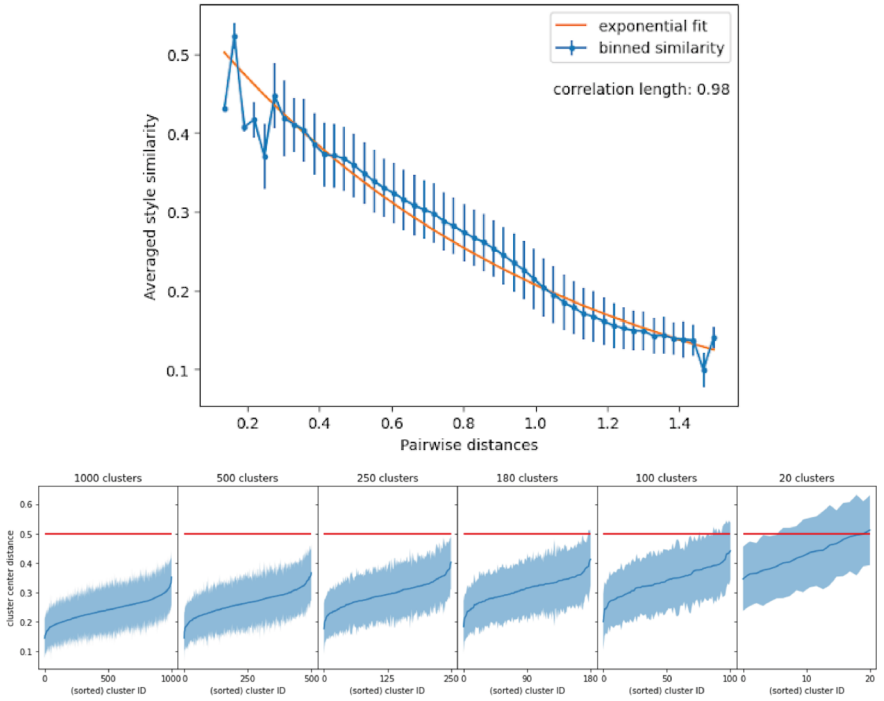


Fig. 2 **Top:** Averaged style similarity over bucketized pairwise distances. An exponential fit is used to determine the decay length of the style-similarity in the embeddings space, which indicates the distance in embedding space at which the style-similarity decays by a factor of $1/e$ **Bottom:** The average and standard deviation of center distances per cluster. The red line indicates the style similarity length scale as determined from a exponential fit on the curve displayed in the top figure

between number and size of segments, as we see clusters clusters being sufficiently smaller than the style-similarity decay length.

Dimensionality reduction

We further conduct experiments to determine whether we can improve the coherence of groups by using a more advanced clustering approach. Due to the scaling behavior of distances in high-dimensional spaces, we apply an additional dimensionality reduction step before applying the k-means algorithm. Additionally, we experiment with the HDBSCAN [32] clustering algorithm, as it has shown good results in combination with UMAP in the topic modeling task [27]. We apply this algorithm in combination with parametric UMAP in the lower-dimensional space. For dimensionality reduction, we experiment with autoencoder networks [36] and UMAP [37]. Due to the size of our datasets, we use the parametric UMAP method [38], which trains a neural network with the UMAP objective as a loss function in order to find a projection onto the low-dimensional representation of the embeddings. This allows for training on a subset and further enables batching of the data, which alleviates

Table 3 Comparison between dimensionality reduction and clustering methods on variant 4 clustered with k-means in different embedding spaces

	Embedding dimension	Silhouette score ↑	ROC AUC ↑
k-means (baseline)	128	0.09 (cosine)	0.71
autoencoder + k-means	5	0.16 (euclidean)	0.65
parametric UMAP + k-means	5	**0.43** (euclidean)	**0.73**
parametric UMAP + HDBSCAN	5	0.14(euclidean)	0.66

hardware limitations. We experiment with three different variants: autoencoder and k-means, parametric UMAP and k-means, and parametric UMAP and HDSCAN, which we compare to the baseline of clustering using k-means directly in the original embedding space. As we are specifically interested if this approach would enable us to find a lower number of highly coherent clusters, we use k-means with $k = 20$. For HDSBAN, the number of clusters cannot be set specifically. Therefore, we tweak the parameters to find an approximately similar number of clusters, resulting in 24 clusters. We follow the evaluation protocol mentioned above and report the style similarity ROC AUC, and the silhouette scores, measured in the low-dimensional embedding space (Table 3). We observe a significant uplift in the silhouette scored as compared to clustering in the embedding space. Specifically, parametric UMAP + k-means shows a nearly five fold increase in this metric. The other two methods also show improvements in this metric, albeit much smaller. Considering the ROC AUC metric, we observe a significant decrease for autoencoder + k-means and parametric UMAP + HDBSCAN. Parametric UMAP + k-means demonstrates a slight improvement over clustering directly in the original embedding space, showing that in this case, the increase in silhouette score also translates to an improvement in the coherence in the clusters.

While we observe an improvement in the clustering metrics with this approach, it also significantly increases the complexity, pipeline runtime, and resource costs. At the current stage, we do not consider the additional costs to be offset by the increase in performance.

Representative segment items

We utilize segmentation of consumers to drive a better experience in recommendations. To this end, we make use of *representative segment items*. These are items that represent the consumers in the given segment. To identify representative segment items, we select the top 100 most popular items by significant actions for each segment and each gender preference. Additionally, we match the distribution of attributes to the distribution of consumer interactions in the segments. This ensures that the representative items are not only popular but also align with the preferences of the consumers in the segment. To boost the significance of the items to the cluster, we only consider events from consumers within a certain radius to the cluster centers when computing the popularity. We also apply business rules to further refine

Fig. 3 The interaction history of example consumers that were identified to belong to one segment (left), the representative segment items identified in their segment (middle), and the fashion expert interpretation of the style (right)

our selection. For example, we account for same items with different colors. This helps to ensure that our recommendations are diverse and cater to the varied tastes of our consumers. Fig. 3 shows the interaction history of example consumers that were identified to belong to one segment (left), the representative segment items identified in their segment (middle), and the fashion expert interpretation of the style (right).

Fashion Expert interpretation

Our fashion expert assessed styles and assignments by evaluating consumer histories and representative segment items. We conducted this evaluation for segmentation using 20, 250, and 1000 clusters. As for the latter two, we sample around 20 segments based on size and average inter-cluster style similarity to make the evaluation feasible. We sampled 15 consumers from each of the selected clusters to be evaluated by a fashion expert. The evaluation concluded that using 20 segments, there was some shared behavior for the consumers closest to the centroid. However, the more distant consumers in the same clusters exhibited weak similarity to consumers close to the centroid, indicating that the segments might be too large. Using 1000 segments, we observe that the segments are over-focused on a single item that is repeated many times throughout histories (e.g., a certain Nike sneaker or a leather belt). The segments in this case seemed to be too specific to carry a notion of style. Of the chose variants, the 250 segments showed the overall best alignment with a notion of style and consistency of consumers. This aligns with our previous investigation on the center-distance distribution.

Recommendations with data-driven segmentation

We aim to utilize the broad group preferences to reduce the cold-start problem occurring in item recommendations for new and low-engaged consumers. To achieve this, we propose and compare three approaches that vary in the way representative segment items are integrated into the recommendations. *Approach 1* replaces individual recommendations entirely with the representative segment items; *Approach 2* backfills a percentage of the consumer's interaction history with representative segment items and uses the resulting sequences as input to the existing algorithms that generates recommendation candidates; *Approach 3* interleaves the resulting output candidates of the existing recommendation algorithms with the representative segment items.

While Approach 1 directly recommends the representative segment items, Approach 2 and 3 aim to maintain a good balance between individual signals from the consumer and the representative items of the segment they belong to. To determine the most suitable approach for style-based item recommendations, we conducted offline evaluations to measure how effectively style-based recommendations can improve the attributes diversity and relevance. To measure the diversity we had a look at the longest sequence of recommendations in the top-k recommendations from the same brand, color, fit, etc. To measure the relevance we used nDCG and the overlap coefficient with historical consumers' interactions. The coefficient also shows how novel the recommendations are compared to the previous consumer history interactions. For nDCG, we use historical consumer click data on the items recommended by existing recommendation algorithms. Additionally, for Approach 2, we measured how much new recommendations differ from the representative segment items in fit, brand, color, etc. This reflects the compatibility between the item recommendation algorithm output and the representative segment items used in the input. Finally, we conducted qualitative visual inspection of example recommendations.

The results indicate that Approach 2 with 20% backfilling of representative segment items had the best performance—increasing brand and commodity group diversity for less engaged consumers, a relatively smaller loss in nDCG, and superior adaptability to consumer context compared to alternative approaches. Visual evaluations of the product recommendation conducted by fashion experts also confirmed that Approach 2 seamlessly combines consumer preferences with representative segment items, resulting in a more diverse product range while still maintaining individual consumer preferences.

We also conducted a detailed analysis of evaluation results for various consumer cohorts. These cohorts were defined based on factors such as the length of a consumer's interaction history, the recency of their expressed interests, and the consistency of their engagement over a 12-week period. The results indicated a decrease in nDCG across approaches and segments, which was anticipated since nDCG is sensitive to item order and does not reward diversity and exploration. Interestingly, we observed an increase in diversity among most approaches and certain consumer cohorts. This led to higher diversity scores for less engaged customers. This finding aligns with our hypothesis that using representative segment items should be more prominent in the recommendations for consumers who have low amounts or no recent signals.

Online testing of recommendations with style groups

We perform an online test of the recommendations with style items reaching 1.2M customers in a period of 2 weeks. To this end, we split the population of each of the segments into a treatment and baseline group. While the treatment is showing hybrid recommendations as described above, the control recommendations are generated by the default hyperpersonalisation model based on a variational autoencoder [39]. This model takes into account both interactions with an item by consumers and content of an item itself.

We report the recommendation performance in terms of consumer engagement with the recommended items which is measured by clicks, wish listing, and

purchasing. Since we are interested analyzing the data-driven grouping impact recommendations and specifically in the case of cold-start problem, we analyze different cohorts of consumers separately. To this end, we split consumers into cohorts by (1) their history length and recency and (2) their distance to the style group centroid.

Overall, we observe that recommendations with data driven segmentation don't surpass the hyper-personalization baseline and decrease the consumer engagement by 0.19%. Looking deeper into the consumer cohorts, we observe a positive impact between 0.1–0.65% across the different groups for consumers with moderate amount of recent interactions (30–120 actions in the past week). Consumers with low or high recent interactions (less than 30 or more than 120 interactions in the past week) show a negative outcome between -0.7–0.95%). In addition, consumers who are grouped closer to the centroids of their respective style group are reacting more positively to the Style Group driven recommendations, as expressed by a 0.6% increase in consumer engagement. On the other hand, the consumers furthest from the centroid of their respective style group respond considerably worse to the treatment as they show a drop of 1.1% in consumer engagement. This is explained by the fact that such consumers are more likely to transition between the different style groups in comparison to the consumers closer to the centriods. This indicates that further improvements on the approach can be made by improving the stability of the embedding and clustering approaches.

4.2 Lookalike Segmentation of Designer Consumers

We explore lookalike segmentation in the context of *designer consumers*. In this use case, we define *core designer consumers* as consumers with a specified minimum amount of interactions with designer brands over the last year, indicating a high affinity towards those brands. Our goal is to find consumer that would likely show a high affinity towards designer brands in the future, in order to present them more relevant recommendations.

Using this core segment definition, we aim to differentiate the historical behavior of core designer consumers from that of non-designers consumers and subsequently identify lookalike designer consumers. Negative consumers were randomly selected from the pool of non-designer consumers, excluding new consumers for whom the first activity on the platform was less than 7 d ago.

Data

We create a dataset composed of timestamped events (article-click and add-to-wishlist) sequences per consumer labeled with the designer status (core or negative). To create training input sequences for the model, we randomly sample 100 consecutive events from the last 4 months of the consumers' histories. We augment the dataset to increase the ratio of positive examples by sampling up to 5 non-overlapping sequences from the core consumers' histories. To create input sequences for inference we take the last 100 events of all consumers not in the core designer set. Additionally,

we utilize consumer specific features, such as sales-channel, age segment and gender preference.

Lookalike model

Using this dataset we train a sequential model based on the transformer architecture [40]. We use the following SKU features: brand, season-code, silhouette-code, tag, material, designer-status, price and whether the brand is followed by the consumer. The features specific to the consumer (age-segment, gender preference, and sales-channel) are provided by the CLS token. For categorical features, we use a trainable embedding layer with dimension 64. Numerical features, i.e. price and event timestamp, are normalized to the unit interval and multiplied by an embedding layer, effectively scaling a trainable embedding vector. For each token, we sum the embedding of each of the features contributing to it. We use a softmax layer for classification, which uses the encoded CLS token as input and outputs scores for both classes—designer and non-designer. The model is trained using a binary-cross entropy loss predicting the designer status of the consumer sequence.

Besides the model described above *Variant 1*, we consider four other variants to further investigate the proposed model; *Variant 2* uses additional data balancing based on class weights, in order equalize the weight of both designers, and non-designers; *Variant 3* uses piece wise linear encoding [41] to represent the numerical features of events (price and timestamp); *Variant 4* combines Variant 3 with data balancing as described in *Variant 2*; *Variant 5* omits the timestamp feature of the events, which essentially removes the order of the events.

Classification threshold optimization

To identify lookalike consumers, we need to set the threshold parameter τ, as defined in Eq. 1. This parameter is a trade-off between the similarity of the lookalikes to the core consumers and the number of lookalikes obtained from the model. We identify the F_2-score as a suitable metric to determine the quality of a set threshold. To find the optimal τ, we compute the F_2-score for a range of threshold and determine their optimum.

Offline experiments

We evaluate five model variants using precision, recall, average precision and F_2-score, the latter of which is our primary evaluation metric. Table 4 shows offline

Table 4 Offline evaluation results for random baseline and the transformer-based classifier on four dataset variants

Variant	F_2-score	Precision	Recall	Average precision
Random	0.087	0.02	0.5	0.02
1	0.414	0.195	0.576	0.263
2	0.412	0.202	0.558	0.265
3	0.417	0.204	0.565	0.27
4	0.415	**0.216**	0.540	0.269
5	**0.439**	0.208	**0.608**	**0.293**

evaluation results for random baseline and the five transformer variants based on a single training and evaluation run. Our model outperforms the random classifier on all the metrics which serves as a basic sanity check. Model Variant 5 shows best performance on the majority of the metrics, achieving F_2-score of 0.439 (+5.3% compared to Variant 3, +6% compared to Variant 1) with the optimized threshold. Precision with optimized threshold is better in Variant 4, but increase in the recall makes up for it when computing the F_2-score.

Figure 4 (left) shows the distribution of scores for ground truth core and non-core consumers on the validation set. Non-core consumers are further divided into ones with zero designer interactions and with one or more designer interactions. These distributions indicate that the model is capable of solving the classification task separating core designer consumers from the rest. Furthermore, there is a large tail of non-core consumers whose score is in the core consumers distribution and all of these consumers have one or more designer interactions, indicating that these consumers already show some level of affinity to designer brands. These consumers are the designer lookalike consumers we are trying to extract. By varying the classification threshold we can extract a different number of lookalike consumers depending on the application. We cannot directly access the quality of these consumers, but we can calculate classification metrics on the validation set using the threshold and treating core consumers as ground truth positive examples. Conversely, we can first optimize the threshold on the validation set and then use it to extract the lookalike consumers.

Figure 4 (right) shows the dependency of the F_2-score and the number of lookalike consumers on the classification threshold. Increasing the threshold reduces the number of consumers classified as lookalikes. Relation with the F_2-score is more

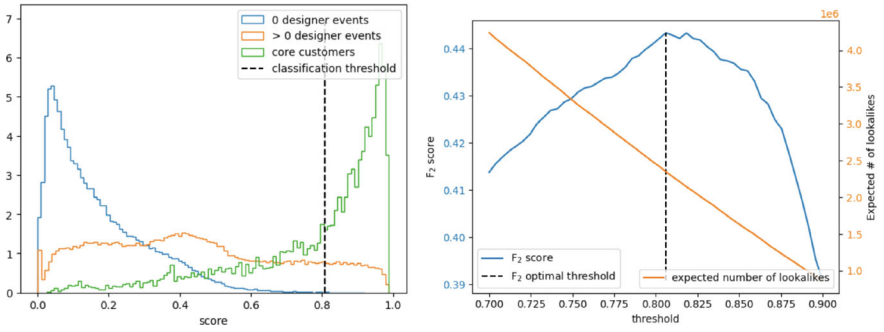


Fig. 4 **Left:** Model predictions for core and non-core consumers on the validation set. Predictions on non-core consumers are further resolved by consumers with zero designer events and consumers that have at least one designer events. The lookalike classification threshold (black dashed line) is determined by maximizing the F_2-score. **Right:** F_2-score and expected number of lookalike consumers depending on the threshold: By varying the classification threshold, we can vary the number of lookalikes. This curve allows for setting a threshold that might fulfill business needs, such as a minimum number of lookalikes, while still monitoring the quality of the lookalikes as indicated by the F_2-score

complex, peaking at the threshold value of 0.808. This threshold value gives us 2.31M of designer lookalike consumers.

Online experiments
We ran an online experiment to test the lookalike Designer consumers on the product catalog page. To this end, we equally split the set of lookalike consumers into a treatment and control group. The two-phased ranking model first fetches the available items along with their attributes and popularity scores and then ranks them based on the consumer preferences by considering their interaction history. For the control group, we use the default ranking model, whereas for the treatment group we use a variant fine-tuned on the *core designer consumers* set which ranks items from designer brands higher in the catalog. The expectation was that by offering a ranking experience tailored for designer, the lookalike consumers would be able to discover more items that resonate with their preferences and their aspiration to become Designer consumers compared to the default general ranker approach. The A/B test ran for around 5 weeks and reached 1.64M lookalike designer consumers. Online KPIs primarily relied on the performance of the ranking system and did not directly measure the performance of the lookalike model itself. As a result, we detected a significant positive increase click through rate (0.45%).

5 Conclusion and Next Steps

In this work, we presented UNICON—a unified consumer segmentation framework capable of driving personalization in fashion e-commerce. Our approach consists of learning dense long-term consumer representations that are then utilized to derive two essential classes of consumer groups: lookalike and data-driven. We demonstrated the capability of this framework using real-world large-scale consumer data to identify *lookalike designer segment*—consumers with high affinity towards premium fashion brands—and *data-driven style groups*—segments of consumers exhibiting long-term similar fashion style. Through offline and online tests, we showed that the identified lookalike consumers display high engagement with designer items and that we have improved their experience. Additionally, our comprehensive experimentation with style segments showed the gains of employing consumer groups in a hybrid personalized recommender system that complements strong individual preferences with wider group interests. Future work consists of investigating methods to improve the consumer embeddings to reflect more robust and longer-term behavior representation by expanding the sequence length in the model, extending the considered interactions time frame, and exploring alternative approaches to better summarize the consumer embedding.

We hope our framework motivates new work that improves e-commerce personalization based on identifying and understanding collective consumer behavior.

Acknowledgements We thank Ellen Scherer for her valuable insights in interpreting fashion style, Mathilde Caron for driving the product vision of this project, and both Marjan Celikik and Evertjan Peer for their contributions in the early stages of this line of work.

References

1. Cao D, He X, Miao L, An Y, Yang C, Hong R (2018) Attentive group recommendation. In: The 41st international ACM SIGIR conference on research & development in information retrieval, pp 645–654
2. Dara S, Chowdary CR, Kumar C (2020) A survey on group recommender systems. J Intell Inf Syst 54(2):271–295
3. Delic A, Masthoff J (2018) Group recommender systems. In: Proceedings of the 26th conference on user modeling, adaptation and personalization, UMAP '18. Association for Computing Machinery, pp 377–378
4. Zhenhua H, Yajun L, Choujun Z, Chen L, Weiwei C, Yunwen C (2021) A novel group recommendation model with two-stage deep learning. IEEE Trans Syst Man Cybern: Syst 52(9):5853–5864
5. Pérez-Almaguer Y, Yera R, Alzahrani AA, Martínez L (2021) Content-based group recommender systems: a general taxonomy and further improvements. Expert Syst Appl 184:115444
6. Sankar A, Wu Y, Wu Y, Zhang W, Yang H, Sundaram H (2020) Groupim: a mutual information maximization framework for neural group recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 1279–1288
7. Zhang S, Zheng N, Wang D (2022) Gbert: pre-training user representations for ephemeral group recommendation. In: Proceedings of the 31st ACM international conference on information & knowledge management, pp 2631–2639
8. Bennett P, White RW, Chu W, Dumais S, Bailey P, Borisyuk F, Cui X (2012) Modeling the impact of short- and long-term behavior on search personalization. In: Proceedings of the 35th annual ACM SIGIR conference (SIGIR 2012). ACM, August 2012
9. Choi SH, Kang S, Jeon YJ (2006) Personalized recommendation system based on product specification values. Expert Syst Appl 31(3):607–616
10. Liang D, Krishnan RG, Hoffman MD, Jebara T (2018) Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 world wide web conference, pp 689–698
11. Song Y, Wang H, He X (2014) Adapting deep ranknet for personalized search. In: Proceedings of the 7th ACM international conference on web search and data mining, WSDM '14, New York, NY, USA, 2014. Association for Computing Machinery, pp 83–92
12. Baltrunas L, Makcinskas T, Ricci F (2010) Group recommendations with rank aggregation and collaborative filtering. In: Proceedings of the fourth ACM conference on recommender systems, RecSys '10, New York, NY, USA, 2010. Association for Computing Machinery, pp 119-126
13. Zhang Z, Zheng X, Zeng DD (2016) A framework for diversifying recommendation lists by user interest expansion. Knowl-Based Syst 105:83–95
14. Castro J, Yera R, Martinez L (2018) A fuzzy approach for natural noise management in group recommender systems. Expert Syst Appl 94:237–249
15. De Pessemier T, Dooms S, Martens L (2014) Comparison of group recommendation algorithms. Multimed Tools Appl 72:2497–2541
16. Zhou Y, Dou Z, Wei B, Xie R, Wen J-R (2021) Group based personalized search by integrating search behaviour and friend network. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 92–101

17. Ondrej K, Michal K, Mária B (2016) Personalized hybrid recommendation for group of users: top-n multimedia recommender. Inf Process Manag 52(3):459–477
18. Ma Q, Wen M, Xia Z, Chen D (2016) A sub-linear, massive-scale look-alike audience extension system a massive-scale look-alike audience extension. In: Fan W, Bifet A, Read J, Yang Q, Yu PS (eds) Proceedings of the 5th international workshop on big data, streams and heterogeneous source mining: algorithms, systems, programming models and applications at KDD 2016. In: Proceedings of machine learning research, vol 53. San Francisco, California, USA, 14 Aug 2016. PMLR, pp 51–67
19. Rahman MM, Kikuta D, Abrol S, Hirate Y, Suzumura T, Loyola P, Ebisu T, Kondapaka M (2023) Exploring 360-degree view of customers for lookalike modeling. arXiv:2304.09105
20. Brito PQ, Soares C, Almeida S, Monte A, Byvoet M (2015) Customer segmentation in a large database of an online customized fashion business. Robot Comput-Integr Manuf 36:93–100
21. Peng Y, Liu C, Shen W (2023) Finding lookalike customers for e-commerce marketing. arXiv:2301.03147
22. Celikik M, Wasilewski J, Mbarek S, Celayes P, Gagliardi P, Pham D, Karessli N, Ramallo AP (2022) Reusable self-attention-based recommender system for fashion. In: Workshop on recommender systems in fashion and retail. Springer, pp 45–61
23. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. CoRR, arXiv:abs/1706.03762
24. Mall U, Matzen K, Hariharan B, Snavely N, Bala K (2019) Geostyle: discovering fashion trends and events. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 411–420
25. Matzen K, Bala K, Snavely N (2017). Streetstyle: exploring world-wide clothing styles from millions of photos. arXiv:1706.01869
26. Kiapour MH, Yamaguchi K, Berg AC, Berg TL (2014) Hipster wars: discovering elements of fashion styles. In: Computer Vision–ECCV 2014: 13th European conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I 13. Springer, pp 472–488
27. Angelov D (2020) Top2vec: distributed representations of topics. arXiv:2008.09470
28. Grootendorst M (2022) Bertopic: neural topic modeling with a class-based TF-IDF procedure. arXiv:2203.05794
29. Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: International conference on machine learning. PMLR, pp 1188–1196
30. Reimers N, Gurevych I (2019) Sentence-BERT: sentence embeddings using siamese BERT-networks. arXiv:1908.10084
31. MacQueen J et al (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol 1. Oakland, CA, USA, pp 281–297
32. Leland MI, John H, Steve A (2017) hdbscan: hierarchical density based clustering. J Open Source Softw 2(11):205
33. McInnes L, Healy J (2017) Accelerated hierarchical density based clustering. In : 2017 IEEE international conference on data mining workshops (ICDMW). IEEE, pp 33–42
34. Wang C, Pan S, Hu R, Long G, Jiang J, Zhang C (2019) Attributed graph clustering: a deep attentional embedding approach. arXiv:1906.06532
35. Rousseeuw Peter J (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20:53–65
36. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation, parallel distributed processing, explorations in the microstructure of cognition, ed. de rumelhart and j. mcclelland. vol. 1. 1986. Biometrika 71:599–607
37. McInnes L, Healy J, Melville J (2018) Umap: uniform manifold approximation and projection for dimension reduction. arXiv:1802.03426
38. Sainburg T, McInnes L, Gentner TQ (2021). Parametric UMAP embeddings for representation and semisupervised learning. Neural Comput 33(11):2881–2907
39. Li X, She J (2017) Collaborative variational autoencoder for recommender systems. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 305–314

40. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems, p 30
41. Yury G, Ivan R, Artem B (2022) On embeddings for numerical features in tabular deep learning. Adv Neural Inf Process Syst 35:24991–25004

AdBooster: Personalized Ad Creative Generation Using Stable Diffusion Outpainting

Veronika Shilova, Ludovic Dos Santos, Flavian Vasile, Gaëtan Racic, and Ugo Tanielian

Abstract In digital advertising, the selection of the optimal item (recommendation) and its best creative presentation (creative optimization) have traditionally been considered separate disciplines. However, both contribute significantly to user satisfaction, underpinning our assumption that it relies on both an item's relevance and its presentation, particularly in the case of visual creatives. In response, we introduce the task of *Generative Creative Optimization (GCO)*, which proposes the use of generative models for creative generation that incorporate user interests, and *AdBooster*, a model for personalized ad creatives based on the Stable Diffusion outpainting architecture. This model uniquely incorporates user interests both during fine-tuning and at generation time. To further improve AdBooster's performance, we also introduce an automated data augmentation pipeline. Through our experiments on simulated data, we validate AdBooster's effectiveness in generating more relevant creatives than default product images, showing its potential of enhancing user engagement.

1 Introduction

Creative optimization seeks to identify the most suitable message for its audience, focusing on the aesthetics and the efficient presentation of the information rather than on the content itself. As such, it complements recommendation systems by

V. Shilova (✉) · L. D. Santos · F. Vasile · G. Racic · U. Tanielian
Criteo, 32 Rue Blanche, 75009 Paris, France
e-mail: veronika.v.shilova@gmail.com

L. D. Santos
e-mail: l.dossantos@criteo.com

F. Vasile
e-mail: f.vasile@criteo.com

G. Racic
e-mail: g.racic@criteo.com

U. Tanielian
e-mail: ugo.tanielian@criteo.com

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_5

addressing not the question of *what to display* to the user, but the question of *how to display* it.

Traditional recommendation literature assumes that the reward received by the recommender system from the user (e.g. click, conversion) is solely due to the quality of the recommendation. This marginalizes out the impact of the creative optimization step, that selects the best image together with its text description to describe the recommended item. We believe this is a missed opportunity since the two steps should be in sync. As an illustrative example, imagine that the user intent is *sport shoes for work*, and that the recommender system returns a very relevant pair of sneakers, but the creative optimization step selects the best non-personalized creative for the product, which is the *pair of shoes worn on a basketball terrain*. It is easy to imagine that the choice of such a creative will affect the final probability of conversion on the said pair of sneakers.

However, such a limitation seems almost insurmontable given the current creative optimization approaches: currently, many of the performance advertising companies employ *Contextual Multiarmed Bandits* as a solution for creative optimization. This approach is particularly fitting in the cases where the range of alternatives for a specific item is limited, usually ranging between 3–10 ad versions and when the number of personalisation contexts is also small.

Very recently, the field of image and text generation witnessed a surge of advances (such as the open release of the Stable Diffusion model [1]), that lead to photorealistic *Text-to-Image Generation* capabilities. In our view, this presents the field of ad creative optimization with a tremendous opportunity to reinvent itself and get itself perfectly aligned with the recommendation step, leading to possibly significant performance improvements.

1.1 Our Proposal

In the domain of digital advertising, one major challenge is to create ads that can resonate with diverse user interests. The creative element of an ad has a profound impact on the user's propensity to engage, making personalized creative content an essential factor in ad success. Existing approaches to ad creation, while effective to some extent, fall short in tailoring the creatives to individual user's interests. Our work attempts to bridge this gap by introducing and formalizing the task of *Generative Creative Optimization (*GCO*)*, a novel concept that leverages user interest signals to personalize creative generation.

We present *AdBooster*, a new model for generating personalized ad creatives based on the Stable Diffusion outpainting model architecture. AdBooster offers a distinctive way to incorporate individual user interest distribution during the fine-tuning process and at creative generation time, thereby ensuring that the creatives generated are specifically tailored to each user's preferences. To the best of our knowledge, this is the first study that integrates a generative model with ad creative personalization based on user interests.

In addition, we propose an innovative automated data augmentation pipeline to enhance the robustness and performance of our model. This pipeline feeds into the fine-tuning process of AdBooster, adding another layer of optimization to the creative generation process.

We carry out a series of experiments to demonstrate the efficacy of our approach. The results showcase an increased relevance of the ad creatives generated by AdBooster compared to the default product images, thereby leading to improved user engagement. The AdBooster model, with its novel Generative Creative Optimization approach, automated data augmentation pipeline, and Stable Diffusion outpainting architecture, holds significant potential in revolutionizing the ad creative generation process.

Contributions:

- Introduce and formulate the task of *Generative Creative Optimization* that personalizes the creative generation using the same user interest signal with the recommendation step, therefore perfectly aligning the two;
- Propose *AdBooster*, a generative ad creative model based on the Stable Diffusion outpainting model architecture that can be fine-tuned on the user interests distribution and be conditioned on the user interests at creative generation time;
- Propose an automated data augmentation pipeline that can be used for fine-tuning the AdBooster model;
- Run extensive experiments that show the increased relevance of generative ad creatives compared to default product images.

2 Related Work

2.1 Creative Optimization

Creative optimization, a vital component in digital advertising, has remained relatively unexplored in machine learning literature, often overshadowed by the extensive research focused on recommendation systems [2, 3]. Creative optimization targets the essential question of 'how' to present the recommended item, which significantly contributes to user engagement and satisfaction [4, 5].

The *Contextual Multiarmed Bandits* method has been widely employed for creative optimization, especially when the creative alternatives for a specific item are limited [6, 7]. This technique adeptly balances exploration and exploitation, optimizing user engagement.

2.2 Text-to-Image Approaches

Recently, with the advent of advanced image and text generation models, such as the Stable Diffusion model [1], new avenues have been opened in creative optimization.

These models, capable of generating photorealistic images from text descriptions, offer opportunities for personalized and dynamic creative generation.

The concept of diffusion probabilistic model (DM), a class of latent variable models inspired by nonequilibrium thermodynamics, was introduced in [8]. This architecture was later improved by Denoising Diffusion Probabilistic Model (DDPM) [9] which offered a different training procedure based on a novel connection between DMs and denoising score matching with Langevin dynamics. Despite the high quality results on image generation, these models had extremely low inference speed. The issue was partially mitigated by using such advanced training and sampling strategies as Denoising Diffusion Implicit Model (DDIM) [10], score-based diffusion [11], and progressive distillation [12]. These architectures primarily use U-net [13] as their backbone neural network.

However, since all these methods operate directly in image pixel space, they still have a significant disadvantage of high training costs. To minimize the computational resources needed for training a diffusion model, inspired by the concept of latent image [14], the Latent Diffusion Model (LDM) approach was introduced in [1]. Over time, this approach was expanded and refined into what is now known as Stable Diffusion.

Furthermore, it is worth noting that the mathematical concepts behind diffusion models, such as Langevin dynamics, score matching, SDEs, etc., make it hard to enter the subject. Therefore, some efforts have been taken in order to reformulate DMs in a simpler way. In [15], the authors derive a minimalist yet powerful deterministic diffusion model that is simple to implement, numerically stable and achieves high quality results in a number of experiments.

Diffusion models have proven themselves effective in text-to-image generation tasks, producing state-of-the-art image generation results. This is typically accomplished by encoding text inputs into latent vectors using pre-trained language models such as CLIP [16]. There is a number of recent successful methods for text-to-image translation: Glide [17] is a text-guided diffusion model that supports both image generation and editing; Stable Diffusion represents a large-scale implementation of latent diffusion aimed at achieving text-to-image generation; Imagen [18] is a text-to-image framework that forgoes latent images and instead directly diffuses pixels using a pyramid structure. Another approach called Dreambooth [19] was built on top of Imagen to "personalization" of text-to-image diffusion models. Given as input a few images of a subject, it is able to generate new photorealistic images of it contextualized in different scenes.

Lately, two novel architectures were introduced in the field of large diffusion models: ControlNet [20] and Composer [21]. They both tackle the problem of limited controllability of large diffusion models, allowing DMs to support additional input conditions, such as edge maps, segmentation maps, keypoints, color histogram and more. By enhancing the control of large diffusion models, these advancements pave the way for broader applications and possibilities.

In this paper, we attempt to show how the recent breakthroughs in generative AI can be leveraged in the field of personalized advertisements generation, taking as an example fashion and retail domain specifically. This area has a lot of

potential challenges and limitations. Ad creative algorithm would need to model complex fashion scenes and diverse background styles, while handling fine details, texture, complex object interactions and semantic consistency in the synthesized backgrounds.

3 Our Approach

In this section, we introduce the formal definitions of recommendation and classical creative optimization and propose a Generative Creative Optimization objective, that uses generative text-to-image models to align recommendation and creative optimization by fully sharing the user context.

3.1 Formal Problem Statement

Recommendation
The training objective for recommendation can be written as:

$$\theta^* = \arg\max_{\theta} \sum_{(u,i)} R(u,i)\pi_\theta(i|u) \tag{1}$$

where $R(u,i)$ is the reward that the user u returns when shown a recommendation for item i and π_θ is the recommendation policy that is represented for generality purposes as a distribution over all possible recommendable items. In most cases the policy will be a Dirac on the best item i^*.

At inference time, for each user u, we select the best item i using the learnt policy:

$$i^* = \arg\max_{i} \pi_{\theta^*}(i|u) \tag{2}$$

Classical Creative Optimization
Creative testing is the way marketers compare the performance of different creatives in a campaign in order to evaluate which creatives yield better results. In practice marketers use two setups for matching creatives with user groups: A/B testing, which is a commonly used method and widely known method, and the multi-armed bandit, which is less commonly known and used, usually due to its relative complexity.

Formally, this can be seen as finding the policy that best matches product image creatives with the users. As a result, the training objective for creative optimization can be written as:

$$\gamma^* = \arg\max_{\gamma} \sum_{(u,i)} R(u,\hat{c}(i))\pi_\gamma(\hat{c}(i)|u,i) \tag{3}$$

where π_γ is the creative optimization policy and $\hat{c}(i) \in C_i$ is one of the ad creatives available for product i.

At inference time, for each user u and the best item i^*, we select the best creative using the creative optimization policy:

$$c^*(i^*) = \arg\max_{c(i^*)} \pi_{\gamma^*}(c(i^*)|u', i^*) \tag{4}$$

Note: In practice, this is not conditioned on the full user context u, but on a low-granularity representation of it, u'. In our proposal, we will be able to condition the creative on the full user representation, allowing us to align the recommendation and creative optimization models.

Generative Creative Optimization
With the advent of photorealistic image generation models, we are not constrained anymore to the task of matching users with a discrete set of pre-existing creative choices and instead are able to generate an unbounded number of potential choices, making it a continuous optimization problem. As a result, the training objective for generative creative optimization task becomes:

$$\gamma^* = \arg\max_{\gamma} \sum_{(u,i)} R(u, G_\gamma(c(i), u)) \tag{5}$$

where G_γ is the creative generation network and $c(i)$ is the default product image (e.g. the product on a simple white background).

At inference time, for each user u and the best item i^*, we generate the best creative using the creative generation network that takes the same user representation u as recommendation:

$$c^*(i^*) = G_{\gamma^*}(c(i^*), u) \tag{6}$$

If we explicitly separate the text and image descriptions of the initial asset, we have that:

$$\gamma^* = \arg\max_{\gamma} \sum_{(u,i)} R(u, G_\gamma(c(i), t(i), u)) \tag{7}$$

3.2 Generative Creative Optimization Using Stable Diffusion Outpainting

In our paper we concentrate on the concrete problem of generating ad creatives using Stable Diffusion outpainting models. Given an outpainting model conditioned on image, text and product mask we have that:

$$\gamma^* = \arg\max_{\gamma} \sum_{(u,i)} R(u, G_\gamma(c(i), T_\phi(i, u), m(i))) \tag{8}$$

where:

- $c(i)$ is the initial product image
- $T_\phi(i, u)$ (resp. $T_\phi(i)$) is a text generative model that takes as inputs both the product and user interest descriptions (resp. the product descriptions) and returns the textual prompt for the image out-painting model
- $m(i)$ is the product image mask

Studying the impact of personalized generation using pretrained text and image generators.

For the rest of the paper we concentrate on studying the impact of contextualizing the generation of the images based on the user interests/type. More explicitly, we are interested in showing that the quality of the generations gets better if contextualized with the full user context versus their un-contextualized counterparts:

$$\sum_{(u,i)} R(u, G_\gamma(c(i), T_\phi(i, u), m(i)) > \sum_{(u,i)} R(u, G_\gamma(c(i), T_\phi(i), m(i)) \quad (9)$$

3.3 Fine-Tuning the Outpainting Model for Fashion-Specific Ad Creative Generation

We need to fine-tune Stable Diffusion (SD) [1] model to be able to use it efficiently for ad creative generation. It is essential, since SD was trained for inpainting task (masks of small size were used), whereas, in our case, we want to outpaint rather large regions. Furthermore, it is crucial in our task that the outpainting model does not modify the product itself. We noticed that without the fine-tuning, SD often extends the frontiers of a product.

A Stable Diffusion [1] model can be decomposed into several key models:

- A text encoder that projects the input prompt to a latent space.
- A variational autoencoder (VAE) that projects an input image to a latent space.
- A diffusion model that refines a latent vector and produces another latent vector, conditioned on the encoded text prompt.
- A decoder that generates images given a latent vector from the diffusion model.

Only the diffusion model parameters are updated during fine-tuning, while the (pretrained) text and the image encoders are kept frozen. Therefore, the fine-tuning loss is calculated between the predicted noise by the diffusion model and the original noise.

To fine-tune SD for background outpainting and then to test our hypothesises on ad creatives generation, we need a large dataset of images with corresponding masks and captions. However, usually such datasets are not available. That is why we propose a data augmentation pipeline which automatically generates product masks and textual prompts from a given dataset of images which can be later used for fine-tuning.

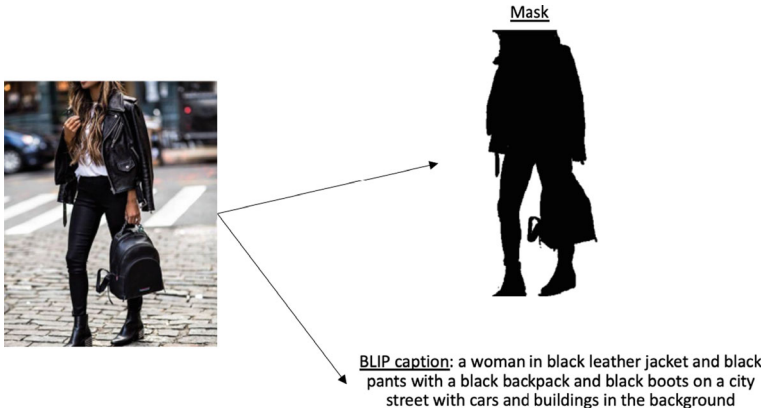


Fig. 1 Example from data augmentation pipeline

The data augmentation pipeline consists of the following steps: (1) data cleaning (see Appendix 7); (2) product mask extraction; (3) prompt generation. To get masks, we use background removal model based on U^2-Net introduced in [22]. To generate prompts, we apply BLIP model to every image in the dataset which was introduced in [23]. Figure 1 gives an example from data augmentation pipeline. The further details are provided in Appendix 7.

It is worth pointing out that it is also possible to add task specific regularization term to the loss function to prevent product extension. For example, after outpainting we can segment the product again and compare its mask with the one used for generation. However, in our experience, this step is not necessary and fine-tuning the model with data augmented pipeline is sufficient to overcome this potential drawback.

The importance of fine-tuning is also confirmed by the decrease of Fréchet Inception Distance (FID) metric [24]. On the dataset which are used in our experiments and which is introduced in Sect. 4.2, we obtained FID of 30.46 for pretrained SD model [1] and FID of 25.73 for the fine-tuned model.

3.4 User Representation and Prompting Strategies

Since one of the key ideas of this paper is to investigate the influence of ad personalization on users, we need to set a user representation first. A user u is defined by a vector $z(u)$ which contains information about user's query $Q(u)$ (description of a product that a user is looking for) and a product context $C(u)$. For example, a user query can be "women red shoes" and its context may be "wedding". Here, we define $z_{QC}(u)$ as the concatenation of $Q(u)$ and $C(u)$ of a given user u, thus in the previous example, $z_{QC}(u) =$ "women red shoes for wedding".

In our approach we generate the new image creative which is contextualized either directly on the user representation $z_{QC}(u)$, either on the textual prompt given by a

text generative model $z_P(u) = T_\phi(i, u)$ which takes as input recommended product i, $Q(u)$ and $C(u)$. We use GPT-3.5 model (gpt-3.5-turbo) [25] from OpenAI API as $T_\phi(i, u)$.

4 Experiments

In our experiments section, we highlight the impact of personalization on the perceived relevance of the items on the user feedback due to the shared user representation.

4.1 Experimental Setup

In order to test our hypothesis we design the following experimental setup:

1. We decide on an open product data set containing both product images and textual descriptions
2. We generate synthetic user representations $z_{QC}(u)$ defined as the concatenation of shopping queries and product context
3. We generate a synthetic data set of (user, item) pairs by returning for every user a random product from the top k most similar products with $z_{QC}(u)$ in the text-based CLIP similarity space. This step aims at simulating a search/recommendation engine logic.

The baselines
The baseline creative for a product is the default catalog image (usually the product on a simple white background). Additionally, since our test product images are actually not a white background we can calculate the uplift in performance versus a second baseline, which is the image of the product in its original background.

Performance metric
In order to compare our proposed approach and the baseline we propose to compute the reward obtained by the user seeing a product image creative as $R(u, i) =< CLIP(z(u)), CLIP(c(i)) >$, which can be seen as the probability of the user buying the item modeled as a function of the affinity between its stated interest and the product image.

In this experimental setup, we investigate two prompting strategies for outpainting: we generate an ad creative image using either (1) $z_{QC}(u)$ or (2) $z_P(u)$. In both cases, the performance metric is calculated with respect to initial user representation $z_{QC}(u)$, since we want to measure the influence of ad contextualization on a given user.

4.2 Data

For our experiments, we chose Pinterest's Shop the Look (STL) Dataset collected by [27]. The dataset contains around 60k scenes of people wearing fashion on different backgrounds ranging from various indoor settings (home, work, gym, shops etc.) to diverse outdoor ones (nature, city setups, etc.). We run STL dataset through the data augmentation described in Sect. 3.3.

Furthermore, for evaluation we generated 30 different user profiles with GPT-3.5 (gpt-3.5-turbo) model [25] from OpenAI API, each containing user's query and context – z_{QC} and corresponding z_P. For the scope of this paper, we chose 4 different categories for backgrounds to experiment with. We mainly focus on seasons (winter, spring, summer, fall); settings (nature vs city setups); various life events (marriage, vacation, parties); and sports (tennis, basketball, hiking).

4.3 Experimental Results

In Tables 1 and 2 we compare two baselines: *BestNeutralImage* which simulates the response rate we obtain by deploying a combination of content-based recommendation and no creative optimization, *BestContextualSetting* which simulates the response rate we obtain by deploying a combination of content-based recommendation and full user-based creative optimization (which is an upper bound of the perf.

Table 1 Performance comparison (CLIP similarity score) between BestNeutralImage, BestContextualSetting, and Generative Creative Optimization (*GCO*) across different contexts: Seasons, Settings, Events, and Sports

			BestNeutralImage baseline	
	BestNeutralImage	BestContextualSetting	*GCO* (z_{QC})	*GCO* (z_P)
Seasons	22.903	21.925	**12.41%** ↑	**10.54%** ↑
Settings	20.11	18.879	**14.68%** ↑	**12.55%** ↑
Events	24.039	22.264	**14.65%** ↑	**10.45%** ↑
Sports	23.816	20.503	**21.5%** ↑	**20.37%** ↑

Table 2 Performance comparison (CLIP similarity score) between BestNeutralImage, BestContextualSetting, and Generative Creative Optimization (*GCO*) across different contexts: Seasons, Settings, Events, and Sports

			BestContextualSetting baseline	
	BestNeutralImage	BestContextualSetting	*GCO* (z_{QC})	*GCO* (z_P)
Seasons	22.903	21.925	**15.56%** ↑	**13.67%** ↑
Settings	20.11	18.879	**15.16%** ↑	**12.96%** ↑
Events	24.039	22.264	**15.34%** ↑	**11.92%** ↑
Sports	23.816	20.503	**27.87%** ↑	**26.94%** ↑

of systems deployed in practice), and GCO, which blends content-based recommendation with a fully personalized creative, powered by our AdBooster model. In our setting, *BestNeutralImage* is an image of a product on a white background (no user context at all), *BestContextualSetting* is an image of the best product recommendation from the catalog (predefined context), and for GCO, it is an image generated via our AdBooster model (user context incorporated).

In our experiments, for each GCO (z_{QC}) and GCO (z_P), we generated 4 different images and kept the one with the maximum CLIP score for display and metrics calculation. Empirically, the images with the largest CLIP scores gave us the best perceptual quality in the most cases. However, it was not always true, and you can find some examples in the Appendix 8.

From the Tables 1 and 2, we observe that scores for *BestContextualSetting* are lower than for *BestNeutralImage* on average. It can be explained by the fact that predefined ad images from a catalog do not always have the background that is aligned with user context. For example, potential user may look for "a yellow dress for the beach vacation", however, there may be no picture of a yellow dress in the catalog that is shot on background with a beach. This issue can be tackled with AdBooster model, since it can tailor background content to users' needs.

Quantitatively speaking, both GCO conditioned on z_{QC} user representation and GCO conditioned on z_P user representation give an uplift of CLIP score over the two baselines. GCO (z_{QC}) gives, on average, a 15% increase in terms of CLIP similarity score across all categories. Moreover, GCO (z_P) also shows an increase in the CLIP similarity score, approximately by 13%.

In terms of qualitative analysis, we can conclude as well that generated backgrounds are closer to users' needs than the predefined ones, so the user is more likely to pay attention to the generated ads. Thus, the use of *GCO* gives us a lot of flexibility regarding to ad creations.

In our experiments, we achieved the best visual results for seasons category. The outpainting model seems to capture almost perfectly "the idea" of natural settings and is able to reproduce them with a high degree of photo-realism. The city setups (buildings, streets) look very realistic as well. You can see some examples in Fig. 2.

However, we still face a number of difficulties that need to be resolved in the future. One of them is that we may occasionally get deduplicated generations (see Appendix 7 for more information). Another minor model's struggle is that it cannot always generate reliably the light on the image or gets confused with ratio of object and subjects presented in the image. Moreover, generating people remains the most difficult task that even fine-tuning cannot resolve with sufficient realism. In the future work, we hope to resolve these problems by using and fine-tuning more powerful version of SD—Stable Diffusion XL [26].

Furthermore, we can compare the generation quality of *GCO* (z_{QC}) against *GCO* (z_P). Even though *GCO* (z_{QC}) gives a slightly greater uplift in terms of CLIP score than *GCO* (z_P), the latter show more variability in the generated background distribution. These generations are often more imaginative and detailed compared to the ones obtained with z_{QC}. All in all, simple concatenation is cheaper to use in practice, as you do not need to use an additional text generative model, and it already gives a

(a) Summer

(b) Hiking

Fig. 2 Examples of AdBooster model generations compared to predefined product images in the seasons and sports categories. Corresponding z_{QC} and z_P are given in Appendix 8

sufficiently great uplift over two baselines. However, using generated prompts gives more flexibility for outpainting, for this reason, it might be interesting to further investigate different ways to do it.

You can find more examples of successful generations as well as some fails in Appendix 8.

5 Conclusions

The present study has made significant strides in the realm of personalized ad creative generation. By introducing the concept of Generative Creative Optimization (*GCO*), we have opened new avenues for ad personalization that specifically target the creative aspect of ads based on user interest signals.

Our proposed model, AdBooster, utilizing the Stable Diffusion outpainting architecture, successfully demonstrates the effective integration of *GCO*, exhibiting its capacity to generate personalized ad creatives. The unique fine-tuning process of AdBooster, which incorporates user interest distribution, is a groundbreaking approach to ad creative personalization.

The innovation of an automated data augmentation pipeline furthers the robustness and efficacy of our model, providing an additional dimension of optimization.

Furthermore, by using a shared user representation for both recommendation and creative optimization, we truly intertwine these tasks, aligning the goals of choosing the optimal item and crafting the ideal creative presentation. This joint

approach creates synergy, enhancing the overall performance and effectiveness of digital advertising.

Through extensive experimental results, we have established the increased relevance of ad creatives generated by AdBooster compared to default product images, subsequently leading to improved user engagement.

In conclusion, our work presents a promising step towards revolutionizing the field of digital advertising. AdBooster, with its focus on personalization through *GCO*, automated data augmentation, and Stable Diffusion outpainting, offers a compelling new method for ad creative generation that is both user-focused and efficient. We anticipate our findings to inspire further research and developments in this direction. As for the future steps, we plan to test Stable Diffusion XL [26] in our experiments to potentially further improve the visual quality of generated ads. We would also like to gather real user feedback for generated ad creatives to get more insight about the visual quality which might possibly be used to enhance *GCO* performance. We also want to try to adopt such architectures as ControlNet [20] and Composer [21] for our *GCO* task to add more control over the ad creatives generation. Moreover, it is possible to investigate different prompting strategies to further boost generation flexibility.

6 Appendix

7 Data Augmentation Pipeline

To further enhance our data augmentation pipeline, we performed data cleaning step after masks' extraction. The idea was that to fully benefit from fine-tuning of SD, the following images should be removed from the dataset: (i) images with simple, monochromatic backgrounds; (ii) images with deduplicated product (several different photos in one—photo collage).

To remove images with monochromatic background, we thresholded on pixels intensities in the following way: (i) convert image to grayscale; (ii) calculate standard deviation of background pixel intensities; (iii) if the standard deviation is below a certain threshold, consider this image to have a monochromatic background.

The standard deviation is a measure of how much the pixel intensities vary from the mean. If the standard deviation is low, it means that the pixel intensities are clustered around a single value, indicating a monochromatic background. The threshold is chosen empirically, in our experiments it equals to 20.

To remove deduplicated product images, we used the mask area in the following way: (i) calculate the ratio of masked area to the whole image area; (ii) if this ratio is above certain threshold, remove the image from the dataset. This approach can also help to filter out too large, non-meaningful, masks which may be occasionally returned by the background removal algorithm. The threshold is chosen empirically, in our experiments it equals to 0.6.

8 Additional Results

8.1 Successful Generations

Figures 2, 3, 4, 5 and 6 give more examples of successful GCO generations. You can find the corresponding z_{QC} (Query, Context) and z_P (Prompt) below.

For Fig. 2a:

- **Query**: yellow summer dress for beach vacation;
- **Context**: planning a trip to Cancun in two weeks and needs a new dress for the beach;
- **Prompt**: A woman with her hair blowing in the wind stands on a sandy beach in a gorgeous yellow summer dress, with turquoise waves crashing behind her and the bright sun shining down, conveying the perfect beach vacation vibe.

For Figs. 2b, 6a:

- **Query**: waterproof hiking boots;
- **Context**: for a weekend hiking trip in the mountains;
- **Prompt**: A rugged pair of waterproof hiking boots sit atop a rocky outcropping overlooking the misty valley below, evoking a sense of adventure and the promise of breathtaking vistas on a weekend trek through the mountains.

For Fig. 3a:

- **Query**: women's long winter coat for snowy weather;
- **Context**: winter business trip to New York City;
- **Prompt**: A stylish woman wearing a long, black winter coat stands in Times Square as snow falls around her, holding a briefcase and looking confidently towards the camera.

For Figs. 3b, 5b:

- **Query**: elegant formal dress;
- **Context**: spring outdoor wedding guest;
- **Prompt**: A woman wearing a beautiful spring formal dress stands in a garden with blooming flowers, as she smiles at the camera as a guest of an outdoor wedding.

For Fig. 3c:

- **Query**: trendy sunglasses;
- **Context**: needed for a beach vacation this summer;
- **Prompt**: A vibrant photo featuring a stylish pair of sunglasses against the backdrop of a picturesque sandy beach with crystal clear turquoise waters and palm trees swaying in the gentle sea breeze.

For Fig. 3d:

- **Query**: black leather jacket;

(a) Winter

(b) Spring

(c) Summer

(d) Fall

Fig. 3 Examples of AdBooster model generations compared to predefined product images in the seasons category

- **Context**: for everyday use in the fall season;
- **Prompt**: A chic black leather jacket paired with distressed denim jeans stands out against the breathtaking backdrop of a colorful autumn forest.

For Fig. 4a:

- **Query**: women's jeans;
- **Context**: fashionable jeans for summer photoshoot in New York city streets;

(a) City setup

(b) City (office) setup

(c) Nature setup

(d) Nature setup

Fig. 4 Examples of AdBooster model generations compared to predefined product images in the settings category

- **Prompt**: A woman wearing stylish summer jeans posing confidently on a busy New York city street with towering skyscrapers and bustling crowds forming a dynamic and stunning background.

(a) Party

(b) Wedding

Fig. 5 Examples of AdBooster model generations compared to predefined product images in the events category

(a) Hiking

(b) Tennis

Fig. 6 Examples of AdBooster model generations compared to predefined product images in the sports category

For Fig. 4b:

- **Query**: comfortable knee-length skirts for summer;
- **Context**: for an office with a business-casual dress code and need breathable skirts for the hot summer weather;

- **Prompt**: Woman in an office setting wearing stylish and comfortable knee-length skirts in various pastel colors, against the backdrop of a bright and airy conference room with large windows and potted plants.

For Fig. 4c:

- **Query**: yellow sunglasses for beach vacation;
- **Context**: upcoming beach vacation, preferably polarized and scratch-resistant to protect eyes from the sun and the saltwater;
- **Prompt**: A vibrant and lively photo featuring a pair of yellow, polarized, scratch-resistant sunglasses well-suited for a beach vacation, set against a breathtaking background of clear blue skies, glistening waves, and soft sandy beaches stretching into the horizon.

For Fig. 4d:

- **Query**: summer shorts for curvy women;
- **Context**: planning a beach vacation and wants comfortable shorts that flatter her curves;
- **Prompt**: A vibrant beach background with crystal blue ocean waves crashing on shore.

For Fig. 5a:

- **Query**: red dress for a cocktail party;
- **Context**: attending a party at a fancy;
- **Prompt**: The photo depicts a stunning woman wearing a bright red dress, standing confidently in front of a luxurious modern lounge with a dazzling city skyline as the backdrop, exuding an air of elegance and sophistication.

For Fig. 6b:

- **Query**: sunglasses for outdoor sports;
- **Context**: wear while playing tennis in the summer;
- **Prompt**: A vibrant and energetic photo captured a player enjoying a game of tennis in the bright summer sun, wearing trendy sunglasses perfectly complementing the colorful backdrop.

8.2 Failed Generations

Figure 7 depicts some occasional failures in *GCO* images: poor quality of people generation, deduplicated images, problems with light casting and object/subject ratio. The corresponding z_{QC} (Query, Context) and z_P (Prompt) can be found below.

For Fig. 7a:

- **Query**: winter coats for women;
- **Context**: wear during the snowy winter season in the city;

(a) Winter

(b) Setting

(c) Setting

(d) Setting

Fig. 7 Examples of failed AdBooster model generations compared to predefined product images in different categories

- **Prompt**: Amidst a snowy cityscape, woman in a winter coat with a glowing streetlamp casting a warm and welcoming light over the scene.

For Fig. 7b:

- **Query**: comfortable hiking pants;
- **Context**: for a weekend hiking trip with friends;

- **Prompt**: A group of friends walk confidently through a picturesque forest trail, clad in comfortable and versatile hiking pants, enjoying the beauty of nature on a perfect weekend getaway.

For Fig. 7c, d:

- **Query**: casual women's pants;
- **Context**: wear to work at the office in the big city;
- **Prompt**: A group of professional women with diverse body shapes and skin tones are posing confidently in a modern, sleek office building lobby, wearing stylish and comfortable work pants that perfectly complement their professional attire.

References

1. Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2022) High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10684–10695
2. Bobadilla J, Ortega F, Hernando A, Gutiérrez A (2013) Recommender systems survey. Knowl-Based Syst 46:109–132
3. Zhang S, Yao L, Sun A, Tay Y (2019) Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv (CSUR) 52:1–38
4. Krajbich I, Armel C, Rangel A (2010) Visual fixations and the computation and comparison of value in simple choice. Nat Neurosci 13:1292–1298
5. Teixeira R, Yamasaki T, Aizawa K (2012) Determination of emotional content of video clips by low-level audiovisual features: a dimensional and categorial experimental approach. Multimed Tools Appl 61:21–49
6. Li L, Chu W, Langford J, Schapire R (2010) A contextual-bandit approach to personalized news article recommendation. In: Proceedings Of The 19th international conference on world wide web, pp 661–670
7. Lu T, Pál D, Pál M (2010) Contextual multi-armed bandits. In: Proceedings Of the thirteenth international conference on artificial intelligence and statistics, pp 485–492
8. Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S (2015) Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning, pp 2256-2265
9. Ho J, Jain A, Abbeel P (2020) Denoising diffusion probabilistic models. Adv Neural Inf Process Syst 33:6840–6851
10. Song J, Meng C, Ermon S (2020) Denoising diffusion implicit models. ArXiv:2010.02502
11. Song Y, Sohl-Dickstein J, Kingma D, Kumar A, Ermon S, Poole B (2020) Score-based generative modeling through stochastic differential equations. ArXiv:2011.13456
12. Salimans T, Ho J (2022) Progressive distillation for fast sampling of diffusion models. ArXiv:2202.00512
13. Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In: Medical image computing and computer-assisted intervention-MICCAI 2015: 18th international conference, Munich, Germany, October 5–9, 2015, proceedings, Part III 18, pp 234–241
14. Esser P, Rombach R, Ommer B (2021) Taming transformers for high-resolution image synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 12873–12883
15. Heitz E, Belcour L, Chambon T (2023) Iterative α-(de)Blending: a minimalist deterministic diffusion model. arXiv:2305.03486

16. Radford A, Kim J, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J et al (2021) Learning transferable visual models from natural language supervision. In: International conference on machine learning, pp 8748–8763
17. Nichol A, Dhariwal P, Ramesh A, Shyam P, Mishkin P, McGrew B, Sutskever I, Chen M (2021) Glide: towards photorealistic image generation and editing with text-guided diffusion models. arXiv:2112.10741
18. Saharia C, Chan W, Saxena S, Li L, Whang J, Denton E, Ghasemipour K, Gontijo Lopes R, Karagol Ayan B, Salimans T et al (2022) Others photorealistic text-to-image diffusion models with deep language understanding. Adv Neural Inf Process Syst 35:36479–36494
19. Ruiz N, Li Y, Jampani V, Pritch Y, Rubinstein M, Aberman K (2023) Dreambooth: fine tuning text-to-image diffusion models for subject-driven generation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 22500–22510
20. Zhang L, Agrawala M (2023) Adding conditional control to text-to-image diffusion models. arXiv:2302.05543
21. Huang L, Chen D, Liu Y, Shen Y, Zhao D, Zhou J (2023) Composer: creative and controllable image synthesis with composable conditions. arXiv:2302.09778
22. Qin X, Zhang Z, Huang C, Dehghan M, Zaiane O, Jagersand M (2020) U2-Net: going deeper with nested U-structure for salient object detection. Pattern Recognit 106:107404
23. Li J, Li D, Xiong C, Hoi S (2022) Blip: bootstrapping language-image pre-training for unified vision-language understanding and generation. In: International conference on machine learning, pp 12888–12900
24. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S (2017) Gans trained by a two time-scale update rule converge to a local nash equilibrium. Adv Neural Inf Process Syst 30
25. Brown T, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A et al (2020) Language models are few-shot learners. Adv Neural Inf Process Syst 33:1877–1901
26. Podell D, English Z, Lacey K, Blattmann A, Dockhorn T, Müller J, Penna J, Rombach R (2023) SDXL: improving latent diffusion models for high-resolution image synthesis. arXiv:2307.01952
27. Kang W, Kim E, Leskovec J, Rosenberg C, McAuley J (2019) Complete the look: scene-based complementary product recommendation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10532–10541

Advanced Algorithms for Visual Learning and User Experience in E-Commerce

Efficient Large-Scale Visual Representation Learning and Evaluation

Eden Dolev, Alaa Awad, Denisa Olteanu Roberts, Zahra Ebrahimzadeh, Marcin Mejran, Vaibhav Malpani, and Mahir Yavuz

Abstract Efficiently learning visual representations of items is vital for large-scale fashion recommendations in e-commerce. In this article we compare several pre-trained efficient backbone architectures, both in the convolutional neural network (CNN) and in the vision transformer (ViT) family. We describe challenges in e-commerce vision applications at scale and highlight methods to efficiently train, evaluate, and serve visual representations. We present ablation studies that evaluate visual representations in several downstream tasks. To this end, we present a novel multilingual text-to-image generative offline evaluation method for visually similar fashion recommendation systems. Finally, we include online results from machine learning systems deployed in production on a large-scale e-commerce platform.

1 Introduction

Large-scale visual representation learning has emerged as a powerful technique for addressing a wide range of computer vision tasks. For e-commerce in particular, techniques have been applied across item recognition, visual search, content moderation, and recommendations [11, 18, 50, 54]. Computer vision plays a critical role in online fashion [12], especially on e-commerce platforms which curate unique, handmade, and vintage items of high visual diversity. It remains a challenge to efficiently retrieve visually similar items from large-scale, diverse collections of images. User

E. Dolev · A. Awad
Etsy, 117 Adams St, Brooklyn, NY, USA
e-mail: edolev@etsy.com

A. Awad
e-mail: aawad@etsy.com

D. Olteanu Roberts (✉)
Independent Researcher, 154 W 15th St, New York, NY, USA
e-mail: d.roberts@vt.edu

Z. Ebrahimzadeh · M. Mejran · V. Malpani · M. Yavuz
Etsy, Brooklyn, NY, USA

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_6

queries of images tend to include photos from real-life or uploaded images taken from another source. However, seller uploaded images tend to have higher quality, better lighting, clear backgrounds, and face-forward angles. The heterogeneous sources create a semantic and visual gap when comparing visual similarity, and high quality visual representations are necessary. Expressive visual representations are learned through complex and costly deep learning architectures, memory-hungry high-dimensional vectors, and carefully crafted training regimens. Furthermore, in search-by-text retrieval, there can be a gap between a text query and the visual imagination for the sought item, especially in visually intensive applications such as fashion or interior design. Advances in text-to-image generative AI can help bridge the gap and the impact of large scale visual models stands to expand across recommender and information retrieval domains. Offline evaluation is of vital importance for effectively learning visual representations and remains a challenge. Qualitatively and quantitatively evaluating visual representations can be subjective and often a ground-truth dataset is not readily available. Visual representations in e-commerce are also evaluated through the benefit they bring to end users, measured by ad clicks or item purchases. We evaluate this benefit offline and use it to guide the representation learning process. Visual representation learning also presents challenges to infrastructure. Applications on mobile devices are latency sensitive and efficient inference is paramount. Furthermore, learning and deploying high quality visual representations at scale can rapidly exceed departmental budgets on cloud costs and energy consumption. Efficiency enables increased adoption.

This article presents methods to tackle these challenges effectively by efficiently designing, learning, and evaluating visual representations. To mitigate for heterogeneity of images and for text-query-to-user-stylistic-intent gap in fashion recommendations, we introduce a novel multilingual text-to-image generative method for visually similar items retrieval and recommendation evaluation.

2 Related Work

Visual representations are used for image-based fashion recommendations [12], visual search [11, 18, 52], and stylistically complementary fashion recommendations [20, 23]. However, learning expressive representations [4] and leveraging them at scale is a costly technical challenge. We can think about efficiency in deep learning along three axes [33]: efficiency in architecture design, efficiency in training, and efficiency in inference and serving. **Architecture design efficiency.** Convolutional Neural Networks (CNN) first employed for visual representations [14, 44, 46] captured local spatial patterns in the image, such as edges and corners. Although vision transformers (ViT) [10] lack the spatial inductive biases of CNN, they outperform CNN when trained with a large enough amount of data and may be more robust to domain shifts [5]. However, large vanilla ViT can be expensive to train and serve and more efficient ViT have emerged [9, 27, 48, 51]. Architectural efficiency is typically tackled through data efficiency [47], clever pooling [51], layer dropping, efficient normalization [24, 26], efficient attention [6, 42, 45], or hybrid CNN-transformer

designs [32, 48]. **Training efficiency.** Learning efficiency can be achieved through label efficiency, transfer learning [16, 22, 31], pretraining through self-supervision [13], or efficient finetuning with compression [17, 40]. Multitask learning [8] is an efficiency inducer as demonstrated across machine learning problems [2, 3, 37, 39, 53]. **Inference and serving efficiency.** Visual representations are included in downstream tasks such as retrieval, ranking and recommendations. Efficient top-k retrieval is typically achieved through approximate nearest neighbors [19]. In downstream rankers, pretrained visual representations can be served as weights within the saved model checkpoint or as features from key-value stores [1, 2]. In tasks such as visual search efficient online inference must be achieved through efficient architectures as well as efficient serving infrastructure [1, 25, 26].

3 Methodology

We compare several readily available pretrained backbones which we finetune under a few different settings and evaluate comprehensively offline and online. We focus on finetuning since pretrained weights are publicly available-an efficiency inducer.

3.1 Efficient Architectures in Visual Representation Learning

The EfficientNet uniformly scales the width, depth, and resolution of the network using a set of fixed scaling coefficients. To upscale we can increase the image size by γ^N, network depth by α^N, and width by β^N, where γ, α, β are determined by grid search. We employ EfficientNetB0, the smallest size model in the EfficientNet family. We choose Base ViT16 as a standard vanilla transformer baseline. Figure 1 in [26] compares the accuracy versus latency tradeoffs for a few vision transformer architectures. We experiment with the EfficientFormer l1 and l3, the data-efficient but high-latency DeiT large [47], and MobileVit [32]. The EfficientFormerl3 scores 2.7 ms for iPhone12 inference with 30M parameters, versus EfficientNetB0 at 1.7 ms with 5M parameters (average in 1000 runs) [26]. The EfficientFormer achieves efficiency through multiple blocks downsampling and only employing attention at the last stage. We visualize attention maps of the EfficientFormer l3 pretrained backbone in Fig. 2. Similarly to other vision transformer architectures [38] we see how different heads learn to focus on different salient parts of the image. To extract the image embedding in CNN models, we replace the final convolutional layer in the backbone and attach a smaller 256-dimensional CNN layer for memory efficiency, followed by a batch normalization layer, a swish activation, and a global average pooling layer which aggregates the convolutional output into the final representation. Then we attach m classification heads, where m is the number of tasks. If X is the output of the penultimate layer of the pretrained backbone, then $X_{embed} = GlobalAvgPool(Swish(BN(Conv_d(X))))$. Finally $y_m = Linear_m(X_{embed})$, where y is the prediction from each of the m tasks

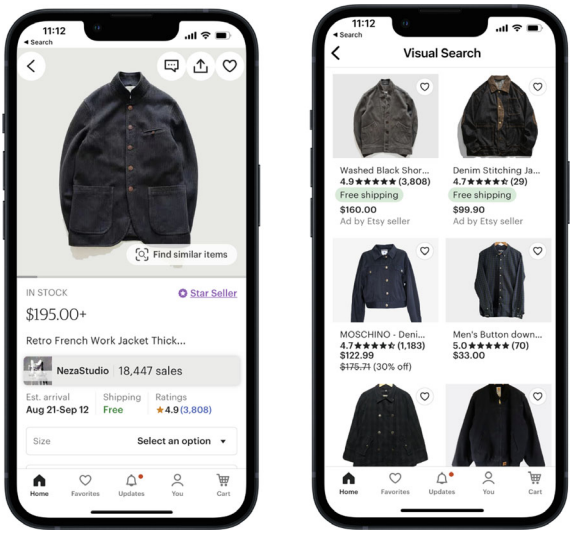


Fig. 1 Visual representations power visual search and visually similar recommendations. A query image (left image) is inferred and the top-k nearest neighbor recommendations are retrieved (right image)

Fig. 2 Probing the EfficientFormer l3 pretrained representations through attention heatmaps. Image of a woman wearing a boho chic dress walking on a narrow street between tall buildings

and $d = 256$. We extract ViT's last hidden state projected down by a dense layer with a *tanh* activation and layer normalization. Then classification heads are added as for the CNN with batch normalization being swapped for layer normalization. When extracting a 512d representation from the EfficientFormer, we average pool the last hidden state and skip the down projection layers.

3.2 Efficient Training

3.2.1 Deep Metric Learning

The predominant approach for learning visual representations that capture similarity is deep metric learning. We employ a three-instance feed-forward network with shared parameters. A positive, negative, and anchor image are fed to the network and triplet loss [15] minimizes the distance between intra-class image representations and maximize the distance between inter-class image representations. The training

dataset includes 10 images for each item. Two images for the same item are considered positive, while an image for any other item is considered negative. A major challenge with deep metric learning is sampling informative triplets. Sampling negatives that are too easy results in slow convergence due to near-zero loss while negatives that are too hard can destabilize training. While larger batch sizes help mining useful negatives, in practice we are limited by hardware memory constraints. Multitask classification can mitigate these challenges.

3.2.2 Multitask Learning

Representations learned using common attributes as multiple supervision signals perform well in diverse downstream tasks, while single-task classification with label the item's taxonomy does not capture visual attributes-colors, shapes, materials. We employ four classification tasks: a top-level taxonomy task with 15 top level categories of the taxonomy tree as labels; a fine-grained taxonomy task, with 1000 fine-grained leaf node item categories as labels; a primary color task; a fine-grained taxonomy task (review photos), where each example is a buyer-uploaded review photo of a purchased item with 100 labels sampled from fine-grained leaf node item categories (datasets details in Table 3). Labels are input by sellers and can be sparse. For this reason we draw samples from an arbitrary set of disjoint datasets as seen in Fig. 3. A data sampler interleaves a uniform number of examples from each dataset into each training minibatch. Each minibatch example has one real label. We set all other labels to the sentinel value -1 and exclude it from loss calculation. A layer rescales, resizes, crops, rotates, and jitters the images at train time. For CNNs, we train new layers from scratch for one epoch with the backbone layers frozen, to avoid excessive computation and overfitting. We then unfreeze 75 layers from the top of the backbone and finetune for nine additional epochs. For efficiency we finetune with a 224×224 resolution since longer sequences from the recommended 384×384 [10] lead to larger training budgets. Most of the ViT output a down-projected 256d representation, finetuned together with the backbone for a constant number of epochs.

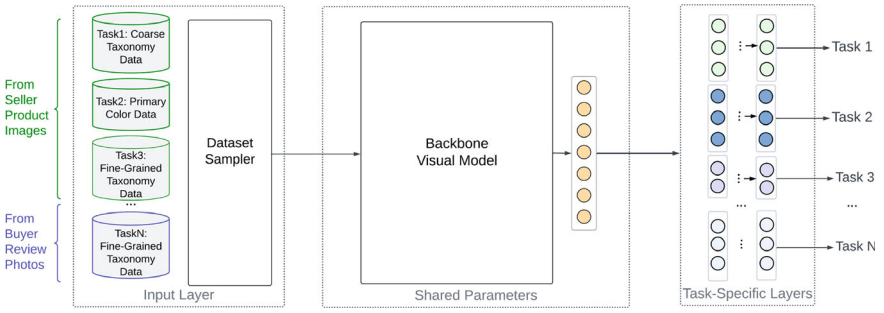


Fig. 3 Multitask visual representations training architecture. The data sampler combines examples from an arbitrary number of datasets corresponding to respective classification heads. The visual representation layer is shared before the heads

3.3 Evaluating Visual Representations for Efficient Learning

Next we present offline evaluation methods and heterogeneous proprietary datasets which can be employed to guide the finetuning of visual representation models toward accuracy on specific downstream tasks, such as visually similar recommendations.

3.4 Offline Retrieval Evaluation Methods

We design an evaluation method to track model performance during and after training on four nearest neighbor retrieval tasks. Each evaluation task has two associated datasets: "queries" and "candidates". The "candidates" dataset feeds a brute force nearest neighbor index, and the "queries" dataset is used to look up K nearest neighbors in the index. The index is constructed on the fly after each epoch to accommodate for embeddings changing. A Keras callback is invoked to compute a Recall@K metric for $K \in \{5, 10\}$ for each task. The "example-wise recall" over N examples is $R@K = \frac{TP_K}{N}$, where the number of true positives, TP is defined as $TP_K = \sum_{i=1}^{N} \left(\mathbb{1}[y_{i,j \in K} = y_{i,\text{query}}] \right)$. Each example that yields at least one true positive in its K candidates is considered a positive example. We can use these Recall@K metrics to efficiently early stop the neural network training. The "Intra-Item" dataset contains groups of seller-uploaded images (several for each item). The query and candidate examples consist of randomly selected images of an item (all seller-uploaded). In the "Intra-Item with Reviews" dataset the query image is a randomly selected buyer-uploaded review image of an item and seller-uploaded images are candidate examples. For both datasets, a candidate image is considered a positive example if it is associated with the same item as the query image. "Visually Similar Ad Clicks" associates seller-uploaded images of items seen on the visually similar surface on mobile with those that have been subsequently clicked. A candidate image is considered a positive example if its associated item has been clicked by a user when viewing the query image. Each evaluation dataset contains 15,000 records for building the index and 5,000 query images to look up the index.

3.5 Offline Evaluation Method Using Efficient Text-to-Image Diffusion Models

Language guided counterfactual test images are used in [36] to stress test image classification models. Building on this concept we propose a novel method for visual representations evaluation using text-to-image diffusion models. From ample historical multilingual text query logs we build a retrieval dataset that bridges the semantic gap between text-based queries and clicked image candidates. A diffusion model

[41, 43] generates high-quality images which become image queries. The "candidates" are images from clicked items corresponding to the source text query in the logs. The source-candidate image pairs form the new evaluation dataset which is then used within the retrieval callbacks. Any given user typing a text query has a different imagination of what that garment might look like, so we generate several images for each query. To mathematically formalize, let us consider a prior $p(\sigma)$, where σ is the intrinsic (unknown) user's fashion style parameter, $\sigma \in S$, and S is the fashion style space. We generate a random sample of length n from the conditional distribution $p(\mathbf{x}, \sigma|q)$ over all possible images $\mathbf{x} \in X$ that can be generated from the seed text query q, for all fashion styles $\sigma \in S$. Then, by Bayes' Rule, the selected image queries are generated by the conditional process $p(\mathbf{x}|q, \sigma) \propto \frac{p(\mathbf{x},\sigma|q)}{p(\sigma|q)}$. For simplicity, we employ the uniform distribution, but other probability distributions can be used. We randomly select one of the n generated images to replace the text query with an image query. Large memory requirements and high inference latency are challenges in employing text-to-image generative models at scale. Efficient text-to-image generation is an area of active research [7, 25, 28]. We employ an open source [35] fast stable diffusion model through token merging [7]. The method provides a 50% speedup of inference, with five times memory consumption reduction, though results depend on the underlying patched model. The query dataset starts with 5,000 text queries and the index dataset has 41,000 examples, with several clicked candidates for each query. A qualitative assessment in Figs. 4 and 5 shows a generated sample for the English text query *"Black bohemian maxi dress with orange floral pattern"*. The generated images include pleasant variations. Interestingly, the fashion models' facial details can suffer some loss while garment patterns remain clear. The diffusion model is multilingual like other transformers in IR [21, 34]. By generating via the French translation of the text query, *"Robe longue noire boheme a motif fleur orange"*, the images display a somewhat French style.

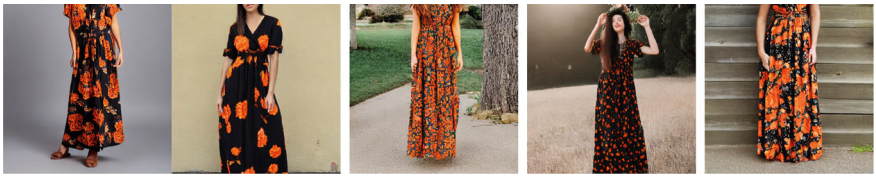


Fig. 4 Text-to-image generation. Random sample of n generated image queries for the English text query *"Black bohemian maxi dress with orange floral pattern"*

Fig. 5 Text-to-image generation. Random sample of n generated image queries for the French text query *"Robe longue noire boheme a motif fleur orange"*

4 Experiments

4.1 Ablation Studies

The experiments aim to answer the research questions: "Which architectures perform best?" "Which training paradigm perform best?". Only the EfficientNetB0 is finetuned in the triplet learning paradigm. All the ViT pretrained weights, publicly available from [49], are finetuned in single-task (ST) classification or multitask with three (MT1) or four tasks (MT2). Additionally, we compare the performance of a linear probe, where we only train the added classification head and keep pretrained weights frozen for the CNN. Projecting down the ViT representation to 256d poses a challenge for evaluation of a vision transformer in the zero-shot paradigm. Thus we employ a pseudo zero-shot method which finetunes all layers on eight examples to lightly update the down-projection layer. Classification accuracy metrics are in Table 1 and retrieval callback metrics are in Table 2. All results are for the 256d representation in all tables, unless otherwise noted. The multitask finetuned ViT Base16 outperforms the other architectures in classification accuracy, but it comes at the cost of high latency, which makes it unusable in latency sensitive applications where the query image is inferred online. The triplet embeddings with EfficientNetB0 win in the evaluation task closest to the metric learning task they were trained on. Even when strong pretrained backbones are available, there is no substitute for a carefully crafted model and finetuning. To evaluate uncertainty, we ran three iterations with different random seeds and we include the upper bound for metrics' standard deviations in Table 2. The numbers for Table 1 are on the same scale. Table 3 gives the hyperparameters for finetuning all models. We evaluated the EfficientNetB0 256d and the EfficientFormerl3 512d representations (finetuned in MT2) using the generated text-to-image queries dataset. On this task, the CNN achieves the best Recall@10 at epoch six (0.0438) and the efficient ViT gets a +2% lift in 12 epochs. This is a challenging evaluation tasks for a few reasons. The clicks are biased toward the production search-by-text system that produced them so absolute values are less relevant. The generative diffusion model displays higher errors than in controlled experiments when employed with actual user queries, which are short and lacking garment detail. Prompt engineering for text queries may improve the process. This is early research which hopefully opens a path for other generative AI-fueled approaches.

4.2 Offline Experiment Results in Additional Downstream Tasks

In Table 4 we compare how several representations perform offline as inputs in two additional downstream tasks beyond visually similar recommendations: an ad click prediction task and a candidate generator for recommendation ads. The baselines are the models without the visual input. The inclusion of EfficientNetB0-based triplet-

Table 1 Classification accuracy metrics. MT1 stands for multitask learning with three tasks, while MT2 has four tasks (MT1+review photos label); ST is single-task classification

Training architecture	Model architecture	Top-level taxonomy		Fine-grained taxonomy		Primary color		Fine-grained taxonomy (reviews)	
		Top1-Acc	Top5-Acc	Top1-Acc	Top5-Acc	Top1-Acc	Top5-Acc	Top1-Acc	Top5-Acc
ST Linear Probe	EfficientNetB0 Backbone	–	–	0.2035	0.4065	–	–	–	–
MT2 Linear Probe	EfficientNetB0 Backbone	0.5084	0.8583	0.2000	0.4171	0.3681	0.7990	0.1581	0.3501
Single-task	EfficientNetB0 Backbone	–	–	0.2883	0.5230	–	–	–	–
MT1	EfficientNetB0 Backbone	0.5980	0.8947	0.2824	0.5174	0.4890	0.8576	–	–
MT2	EfficientNetB0 Backbone	0.6206	0.8992	0.2905	0.5292	0.5040	0.8618	0.2137	0.4305
Pseudo Zero-Shot	ViT Base16 Backbone	–	–	0.0007	0.0052	–	–	–	–
ST	ViT Base16 Backbone	–	–	0.3606	0.6239	–	–	–	–
MT1	ViT Base16 Backbone	0.6760	0.9250	0.3560	0.6120	**0.5800**	0.8900	–	–
MT2	ViT Base16 Backbone	**0.6800**	**0.9300**	**0.3700**	**0.6300**	0.5700	**0.8900**	**0.2700**	**0.5300**
ST	DeiT Backbone	–	–	0.2590	0.5187	–	–	–	–
MT2	DeiT Backbone	0.6693	0.9215	0.3441	0.6006	0.5657	0.8866	0.2603	0.5048
ST	MobileVit xSmall Backbone	–	–	0.2303	0.4594	–	–	–	–
MT2	MobileVit xSmall Backbone	0.6023	0.8955	0.2577	0.4912	0.5242	0.8752	0.1876	0.3942
MT2	EfficientFormerl1 Backbone 256	0.6203	0.9006	0.2981	0.5305	0.523	0.8682	0.2175	0.4398
MT2	EfficientFormerl1 Backbone 448	0.6203	0.9000	0.2981	0.5301	0.5230	0.8882	0.2175	0.4398
ST	EfficientFormerl3 Backbone 256	–	–	0.3018	0.5298	–	–		
ST	EfficientFormerl3 Backbone 512	–	–	0.3076	0.5561	–	–	–	–
MT2	EfficientFormerl3 Backbone 256	0.6345	0.9037	0.3128	0.5458	0.5169	0.8600	0.2332	0.4583
MT2	EfficientFormerl3 Backbone 512	0.6425	0.9080	0.3079	0.5457	0.5509	0.8804	0.2269	0.4477

learned embedding leads to the largest AUC metrics lift in the CTR prediction. In the retrieval task, all finetuned variants perform similarly, with EfficientNetB0 M2 outperforming others slightly in Recall@1000. However, we have not evaluated the EfficientFormer-learned representations in these downstream tasks yet.

Table 2 Recall@5 and Recall@10 metrics and standard deviations for three different similarity tasks. MT1 stands for multitask classification with three tasks, while MT2 has four tasks; ST is single-task classification

Training architecture	Model architecture	Item-item retrieval		Review-item retrieval		Visually-similar clicks	
		R@5	R@10	R@5	R@10	R@5	R@10
Triplet	Pretrained EfficientNetB0 Backbone	0.6963 (0.002)	0.7195 (0.002)	0.2204 (0.001)	0.2566 (0.001)	0.6565 (0.002)	0.7248 (0.002)
Triplet	Finetuned EfficientNetB0 Backbone	**0.8421** (0.005)	**0.8592** (0.005)	0.3539 (0.001)	0.4069 (0.002)	0.6875 (0.002)	0.7562 (0.004)
Zero-Shot	EfficientNetB0 Backbone	0.7226 (0.004)	0.7527 (0.004)	0.2781 (0.001)	0.3174 (0.001)	0.7150 (0.004)	0.7839 (0.004)
ST Linear Probe	EfficientNetB0 Backbone	0.7258 (0.004)	0.7565 (0.004)	0.2870 (0.001)	0.3335 (0.001)	0.7375 (0.004)	0.8104 (0.005)
MT2 Linear Probe	EfficientNetB0 Backbone	0.7335 (0.004)	0.7661 (0.004)	0.3039 (0.001)	0.3550 (0.001)	0.7526 (0.004)	0.8260 (0.005)
ST	EfficientNetB0 Backbone	0.7272 (0.004)	0.7619 (0.005)	0.2781 (0.001)	0.3321 (0.001)	0.7210 (0.004)	0.7956 (0.005)
MT1	EfficientNetB0 Backbone	0.7573 (0.004)	0.7890 (0.004)	0.3021 (0.001)	0.3545 (0.001)	0.7439 (0.004)	0.8182 (0.005)
MT2	EfficientNetB0 Backbone	0.7674 (0.004)	0.7994 (0.005)	0.3455 (0.001)	0.3990 (0.002)	0.7691 (0.004)	**0.8456** (0.005)
Pseudo Zero-Shot	ViT Base16 Backbone	0.6903 (0.003)	0.7208 (0.004)	0.2506 (0.001)	0.2989 (0.001)	0.5883 (0.002)	0.6577 (0.002)
ST	ViT Base16 Backbone	0.7035 (0.003)	0.7378 (0.003)	0.2925 (0.001)	0.3493 (0.001)	0.6467 (0.003)	0.7223 (0.002)
MT1	ViT Base16 Backbone	0.7435 (0.004)	0.7758 (0.004)	0.3140 (0.001)	0.3745 (0.001)	0.6311 (0.002)	0.7056 (0.002)
MT2	ViT Base16 Backbone	0.7634 (0.004)	0.7938 (0.004)	0.3598 (0.002)	0.4240 (0.002)	0.6860 (0.003)	0.7670 (0.003)
ST	DeiT Backbone	0.7332 (0.003)	0.7659 (0.003)	0.3291 (0.001)	0.3870 (0.001)	0.6922 (0.003)	0.7717 (0.004)
MT2	DeiT Backbone	0.7697 (0.004)	0.7956 (0.005)	0.3635 (0.001)	0.4250 (0.002)	0.6900 (0.004)	0.7641 (0.004)
ST	MobileVit xSmall Backbone	0.6376 (0.003)	0.6786 (0.003)	0.2239 (0.001)	0.2739 (0.001)	0.5886 (0.004)	0.6708 (0.004)
MT2	MobileVit xSmall Backbone	0.7241 (0.004)	0.7597 (0.004)	0.2984 (0.001)	0.3604 (0.001)	0.6770 (0.003)	0.7501 (0.004)
MT2	EfficientFormerl1 Backbone 256	0.7568 (0.004)	0.7866 (0.005)	0.3515(0.001)	0.409(0.001)	0.736(0.003)	0.8075(0.005)
MT2	EfficientFormerl1 Backbone 448	0.7929 (0.005)	0.8139 (0.005)	0.3840 (0.001)	0.4391 (0.002)	0.76631(0.004)	0.8345(0.005)
ST	EfficientFormerl3 Backbone 256	0.7086 (0.004)	0.7398 (0.004)	0.2977(0.001)	0.3889(0.001)	0.6831(0.002)	0.7582(0.004)
ST	EfficientFormerl3 Backbone 512	0.7642 (0.003)	0.7904 (0.004)	0.3364 (0.001)	0.3892 (0.001)	0.7232(0.004)	0.7938(0.004)
MT2	EfficientFormerl3 Backbone 256	0.7694(0.005)	0.8007(0.005)	0.3651(0.001)	0.4242(0.002)	0.7212(0.002)	0.7987(0.005)
MT2	EfficientFormerl3 Backbone 512	0.8053 (0.005)	0.8291 (0.005)	**0.4116** (0.002)	**0.4659** (0.002)	**0.78**(0.005)	0.84 (0.005)

Table 3 Finetuning hyperparameters. MT stands for "multitask"

Config	Triplet EfficientNetB0	MT EfficientNetB0	MT ViT B16	MT EfficientFormerl3	MT EfficientFormerl1
Optimizer	Adam	Adam	Adam	AdamW [30]	AdamW
Adam β_1	0.9	0.9	0.9	0.9	0.9
Adam β_2	0.999	0.999	0.999	0.999	0.999
Adam ϵ	1e-08	1e-08	1e-07	1e-07	1e-07
Weight decay	0.00	0.00	0.01	0.05	0.01
Average step Time	21 ms	200 ms	76 ms	37 ms	18 ms
Batch size (train)	512	256	64	128	64
Batch size (validation)	256	256	128	128	128
Input size	$224 \times 224 \times 3$	$224 \times 224 \times 3$	$224 \times 224 \times 3$	$224 \times 224 \times 3$	$224 \times 224 \times 3$
Machine type (training)	2 P100	2 P100	2 P100	2 P100	2 P100
Machine type (validation)	2 P100	2 P100	2 P100	2 P100	2 P100
Learning rate	0.0001	0.0001	0.0001	0.0008	0.0002
Loss	Triplet	CE	CE	CE	CE
Learning rate Scheduler	Cosine [29]	Cosine	Polynomial	Cosine	Polynomial
Num epochs	10	10	5	12	10
Num parameters New Layers (trainable)	737,792	1,260,014	719,598	653,038	637,100
Num parameters (trainable)	2,977,472	3,888,234	87,109,870	31,033,550	12,029,094
Num parameters (total)	4,373,155	4,895,889	87,111,918	31,101,586	12,063,415
Triplet margin	0.2	–	–	–	–
Num train examples	802,822	715,440	715,440	715,440	715,440
Num validation examples	91,576	78,000	78,000	78,000	78,000

Table 4 Ablation studies showing offline % lift for various pretrained visual representations used in the downstream click prediction task in ad ranking (CTR), and candidate generation for recommendation ads (AIR)

Training architecture	Model architecture	CTR		IR	
		ROC-AUC	PR-AUC	R@100	R@1000
No visual inputs	No visual inputs	–	–	–	–
Triplet	EfficientNetB0 backbone	**+0.64%**	**+2.04%**	**+7.58%**	+6.91%
Multitask	EfficientNetB0 backbone	+0.56%	0.0%	+5.93%	**+6.98%**
Multitask	ViT Base16 backbone	+0.62%	+1.9%	+4.38%	+6.05%
Pseudo zero-shot	ViT Base16 backbone	+0.32%	+0.47%	+0.09%	+0.75%

Table 5 Statistically significant online A/B experiment results (p-value ≤ 0.001) using visual representations in e-commerce applications

Online metric	Ad information retrieval	Ad similar items recommendations	Visually similar recommendations		Sponsored-search ranking	
	Ad recommendations matching	Ad recommendations ranker	iOS	Android	CTR	PCCVR
Ad-recommendation clicked	+3.97%	+6.25%	+0.73%	+0.11%	–	–
Ad-Sitewide-CTR	+0.97%	+0.77%	−0.65%	+0.38%	+2.55%	−0.02%
Ad-Sitewide-Clicks	+1.05%	+0.70%	+0.73%	+1.61%	+2.07%	−0.48%
Ad-return-on-spend	-0.40%	+0.31%	+1.92%	–	+1.02%	+3.51%
Ads-post-click purchase rate	–	–	–	+1.18%	+0.29%	+1.26%

4.3 Online Experiments

To further evaluate the quality of the representations, we A/B test EfficientNetB0 MT2 across visual applications at Etsy with results in Table 5. The details of the online experimentation platform are out of scope but in line with standard A/B testing. EfficientFormerl3 MT2 led to a further +0.65% lift in CTR and purchase rate in a first visually similar ads experiment. In Table 5 we can see lift in metrics when including visual representations in sponsored search post-click conversion rate prediction [1] and ranking functions, as well as in learning other embeddings for candidate generation. The EfficientNetB0 MT2 representations first powered visually similar ad recommendations in Fig. 1, as well as the first search-by-image shopping

experience on the same e-commerce platform. Users search using photos taken with their mobile phone's camera and the query image embedding is inferred efficiently online.

5 Conclusion

In this article we present a holistic approach to learning and evaluating image representations in an efficient manner and the benefits derived by employing these representations in downstream tasks, such as visually similar recommendations. We contrast backbones and transfer learning methods. We present four offline evaluation tasks, including a novel text-to-image generative method. We show ablation studies and offline and online results in multiple downstream tasks. Learning visual representations is of paramount importance in visually rich e-commerce and online fashion applications. Doing so efficiently is a challenging goal made possible by advances in the field of efficient deep learning in computer vision.

References

1. Awad A, Roberts D, Dolev E, Heyman A, Ebrahimzadeh Z, Weil Z, Mejran M, Malpani V, Yavuz M (2023) Adsformers: personalization from short-term sequences and diversity of representations in etsy ads
2. Beal J, Wu, HY, Park, DH, Zhai A, Kislyuk D (2022) Billion-scale pretraining with vision transformers for multi-task visual representations. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 564–573
3. Bell S, Liu Y, Alsheikh S, Tang Y, Pizzi E, Henning M, Singh K, Parkhi O, Borisyuk F (2020) Groknet: unified computer vision model trunk and embeddings for commerce. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, pp 2608–2616. https://dl.acm.org/doi/abs/10.1145/3394486.3403311
4. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
5. Bhojanapalli S, Chakrabarti A, Glasner D, Li D, Unterthiner T, Veit A (2021) Understanding robustness of transformers for image classification. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10231–10241
6. Bolya D, Fu CY, Dai X, Zhang P, Hoffman J (2022) Hydra attention: efficient attention with many heads. In: European conference on computer vision, pp 35–49. Springer
7. Bolya D, Hoffman J (2023) Token merging for fast stable diffusion. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 4598–4602
8. Caruana R (1997) Multitask learning. Mach Learn 28:41–75
9. Dong X, Bao J, Chen D, Zhang W, Yu N, Yuan L, Chen D, Guo B (2022) CSWin transformer: a general vision transformer backbone with cross-shaped windows. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 12124–12134
10. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, et al (2020) An image is worth 16×16 words: transformers for image recognition at scale. In: International conference on learning representations
11. Du M, Ramisa A, KC AK, Chanda S, Wang M, Rajesh N, Li S, Hu Y, Zhou T, Lakshminarayana N, et al (2022) Amazon shop the look: a visual search system for fashion and home. In:

Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining, pp 2822–2830. https://dl.acm.org/doi/abs/10.1145/3534678.3539071
12. Elsayed S, Brinkmeyer L, Schmidt-Thieme L (2022) End-to-end image-based fashion recommendation. In: Workshop on recommender systems in fashion and retail. Springer, pp 109–119
13. He K, Chen X, Xie S, Li Y, Dollár, P, Girshick R (2022) Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF conference on computer cision and pattern recognition, pp 16000–16009
14. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
15. Hoffer E, Ailon N (2015) Deep metric learning using triplet network. In: Similarity-based pattern recognition: third international workshop, SIMBAD 2015, Copenhagen, Denmark, October 12–14, 2015. Proceedings, vol 3. Springer, pp 84–92
16. Hoffman J, Rodner E, Donahue J, Darrell T, Saenko K (2013) Efficient learning of domain-invariant image representations
17. Hu EJ, Wallis P, Allen-Zhu, Z, Li Y, Wang S, Wang L, Chen W, et al (2021) Lora: low-rank adaptation of large language models. In: International conference on learning representations
18. Jing Y, Liu D, Kislyuk D, Zhai A, Xu J, Donahue J, Tavel S (2015) Visual search at pinterest. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1889–1898. https://dl.acm.org/doi/abs/10.1145/2783258.2788621
19. Johnson J, Douze M, Jégou H (2019) Billion-scale similarity search with GPUs. IEEE Trans Big Data 7(3):535–547
20. Kang WC, Kim E, Leskovec J, Rosenberg C, McAuley J (2019) Complete the look: scene-based complementary product recommendation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10532–10541 (2019)
21. Lavi D (2021) Learning to match job candidates using multilingual bi-encoder bert. In: Proceedings of the 15th ACM conference on recommender systems, pp 565–566
22. Li C, Yang J, Zhang P, Gao M, Xiao B, Dai X, Yuan L, Gao J (2021) Efficient self-supervised vision transformers for representation learning. In: International conference on learning representations
23. Li E, Kim E, Zhai A, Beal J, Gu K (2020) Bootstrapping complete the look at Pinterest. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, pp 3299–3307. https://dl.acm.org/doi/abs/10.1145/3394486.3403382
24. Li Y, Hu J, Wen Y, Evangelidis G, Salahi K, Wang Y, Tulyakov S, Ren J (2022) Rethinking vision transformers for mobilenet size and speed
25. Li Y, Wang H, Jin Q, Hu J, Chemerys P, Fu Y, Wang Y, Tulyakov S, Ren J (2023) Snapfusion: text-to-image diffusion model on mobile devices within two seconds
26. Li Y, Yuan G, Wen Y, Hu J, Evangelidis G, Tulyakov S, Wang Y, Ren J (2022) Efficientformer: vision transformers at mobilenet speed. Adv Neural Inf Process Syst 35:12934–12949
27. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B (2021) Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10012–10022
28. Liu Z, Feng R, Zhu K, Zhang Y, Zheng K, Liu Y, Zhao D, Zhou J, Cao Y (2023) Cones: concept neurons in diffusion models for customized generation
29. Loshchilov I, Hutter F (2016) Sgdr: stochastic gradient descent with warm restarts. In: International conference on learning representations
30. Loshchilov I, Hutter F (2018) Decoupled weight decay regularization. In: International conference on learning representations
31. Luo Z, Zou Y, Hoffman J, Fei-Fei LF (2017) Label efficient learning of transferable representations acrosss domains and tasks. Adv Neural Inf Process Syst 30
32. Mehta S, Rastegari M (2021) MobileViT: light-weight, general-purpose, and mobile-friendly vision transformer. In: International conference on learning representations
33. Menghani G (2023) Efficient deep learning: a survey on making deep learning models smaller, faster, and better. ACM Comput Surv 55(12):1–37. https://dl.acm.org/doi/abs/10.1145/3578938

34. Olteanu Roberts DA (2021) Multilingual evidence retrieval and fact verification to combat global disinformation: the power of polyglotism. In: European conference on information retrieval, pp 359–367
35. von Platen P, Patil S, Lozhkov A, Cuenca P, Lambert N, Rasul K, Davaadorj M, Wolf T (2022) Diffusers: state-of-the-art diffusion models. https://github.com/huggingface/diffusers
36. Prabhu V, Yenamandra S, Chattopadhyay P, Hoffman J (2023) Lance: stress-testing visual models by generating language-guided counterfactual images
37. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I et al (2019) Language models are unsupervised multitask learners. OpenAI blog 1(8):9
38. Raghu M, Unterthiner T, Kornblith S, Zhang C, Dosovitskiy A (2021) Do vision transformers see like convolutional neural networks? Adv Neural Inf Process Syst 34:12116–12128
39. Roberts D (2019) Neural networks for lorenz map prediction: a trip through time
40. Roberts DA, Roberts LR (2020) Qr and lq decomposition matrix backpropagation algorithms for square, wide, and deep–real or complex–matrices and their software implementation
41. Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2022) High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10684–10695
42. Ryali C, Hu YT, Bolya D, Wei C, Fan H, Huang PY, Aggarwal V, Chowdhury A, Poursaeed O, Hoffman J et al (2023) Hiera: a hierarchical vision transformer without the bells-and-whistles (2023)
43. Saharia C, Chan W, Saxena S, Li L, Whang J, Denton EL, Ghasemipour K, Gontijo Lopes R, Karagol Ayan B, Salimans T et al (2022) Photorealistic text-to-image diffusion models with deep language understanding. Adv Neural Inf Process Syst 35:36479–36494
44. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4510–4520
45. Shaker A, Maaz M, Rasheed H, Khan S, Yang MH, Khan FS (2023) Swiftformer: efficient additive attention for transformer-based real-time mobile vision applications
46. Tan M, Le Q (2019) EfficientNet: rethinking model scaling for convolutional neural networks. In: International conference on machine learning, pp 6105–6114. PMLR
47. Touvron H, Cord M, Douze M, Massa F, Sablayrolles A, Jégou H (2021) Training data-efficient image transformers & distillation through attention. In: International conference on machine learning. PMLR, pp 10347–10357
48. Vasu PKA, Gabriel J, Zhu J, Tuzel O, Ranjan A (2023) Fastvit: a fast hybrid vision transformer using structural reparameterization
49. Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, et al (2019) Huggingface's transformers: state-of-the-art natural language processing
50. Yang F, Kale A, Bubnov Y, Stein L, Wang Q, Kiapour H, Piramuthu R (2017) Visual search at eBay. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery & data mining, pp 2101–2110
51. Yu W, Luo M, Zhou P, Si C, Zhou Y, Wang X, Feng J, Yan S (2022) Metaformer is actually what you need for vision. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10819–10829
52. Zhai A, Kislyuk D, Jing Y, Feng M, Tzeng E, Donahue J, Du YL, Darrell T (2017) Visual discovery at Pinterest. In: Proceedings of the 26th international conference on world wide web companion, pp 515–524
53. Zhai A, Wu HY, Tzeng E, Park DH, Rosenberg C (2019) Learning a unified embedding for visual search at pinterest. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp 2412–2420
54. Zhang Y, Pan P, Zheng Y, Zhao K, Zhang Y, Ren X, Jin R (2018) Visual search at Alibaba. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp 993–1001. https://dl.acm.org/doi/abs/10.1145/3219819.3219820

Diversify and Conquer: Bandits and Diversity for an Enhanced E-commerce Homepage Experience

Sangeet Jaiswal, Korah T. Malayil, Saif Jawaid, and Sreekanth Vempati

Abstract In the realm of e-commerce, popular platforms utilize widgets to recommend advertisements and products to their users. However, the prevalence of mobile device usage on these platforms introduces a unique challenge due to the limited screen real estate available. Consequently, the positioning of relevant widgets becomes pivotal in capturing and maintaining customer engagement. Given the restricted screen size of mobile devices, widgets placed at the top of the interface are more prominently displayed and thus attract greater user attention. Conversely, widgets positioned further down the page require users to scroll, reducing visibility and subsequent lower impression rates. Therefore it becomes imperative to place relevant widgets on top. However, selecting relevant widgets to display is a challenging task as the widgets can be heterogeneous, widgets can be introduced or removed at any given time from the platform. In this work, we model the vertical widget reordering as a contextual multi-arm bandit problem with delayed batch feedback. The objective is to rank the vertical widgets in a personalized manner. We present a two-stage ranking framework that combines contextual bandits with a diversity layer to improve the overall ranking. We demonstrate its effectiveness through offline and online A/B results, conducted on proprietary data from Myntra, a major fashion e-commerce platform in India.

S. Jaiswal (✉) · K. T. Malayil · S. Jawaid · S. Vempati
Myntra Designs Pvt Ltd., Bangalore, India
e-mail: sangeet.jaiswal@myntra.com

K. T. Malayil
e-mail: korah.malayil@myntra.com

S. Jawaid
e-mail: saif.jawaid@myntra.com

S. Vempati
e-mail: sreekanth.vempati@myntra.com

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_7

1 Introduction

In the world of e-commerce, the website or app real estate holds significant value and serves as a key catalyst in shaping user experience, engagement, and conversion rates. Most homepages have distinct sections, comprising various widgets that feature diverse content such as product recommendations, monetization products and ads, promotions, and sale details. As the number of widgets increases, the demand for a systematic and relevant display order becomes more pronounced. Our data reveals that, on average, users scroll down the homepage up to three times and click on widgets within the first two scrolls.

Addressing this challenge posed a few hurdles in our context, Widgets are owned by different teams, and slotting them properly requires domain knowledge and validation through A/B tests, which is time-consuming and also not feasible sometimes. The notion of a widget ID is not tangible since the same widget can be represented with different widget IDs in different layouts. Widgets are also heterogeneous, for example, they can be banners, awareness campaigns, product carousels, engagements (videos), order tracking cards, etc. The contents of these widgets are powered by different recommendation models and are personalized to the user. Hence the representation of these widgets needs to be learned without considering the content. Specifically, the particular brands or products within these widgets are not available as signals to the ranker due to system constraints.

For this reason, we have engineered new features such as Core theme, Widget intent, etc. to identify what type of content is being showcased through the widgets. These have been instrumented as mandatory fields to be filled by business teams while creating/deploying new widgets and layouts.

Existing approaches to tackle ranking problems like this rely on having sufficient historical data [5, 18] to predict user preferences. However, a limitation arises when there is minimal data about a new content item, hindering effective recommendations. This issue, known as the "item cold-start problem" arises especially when introducing novel item types to a platform. To overcome the item cold-start problem and provide effective recommendations, the Multi-Armed Bandit (MAB) approach [21] is proposed. MAB balances the exploitation of known user preferences with the exploration of new items. Contextual bandit algorithms, a type of MAB, utilize context information, including user and item representations, to achieve improved performance. By assuming a linear relationship between the expected reward and context, contextual bandit algorithms can enhance recommendations in cold-start scenarios with limited data about newly added items [1, 6].

Contributions. In this research paper, we utilize the contextual bandit framework to tackle the challenge of optimizing the order of widgets to enhance the customer experience on our mobile shopping app. Our contributions include:

- Implementing a contextual bandit framework with diversity for dynamic widget reordering in real-time.
- Demonstrate superiority of bandit learning in a dynamic environment by comparing the Linear UCB model against a more complex supervised XGBoost model (Fig. 1).

Fig. 1 Example of widgets and layout on our home page: a different set of users can be assigned to different layouts, each with its own set of widgets that may not be mutually exclusive

2 Related Work

Ever since multi-armed bandits [21] were introduced, there has been substantial research and industry implementation in various domains such as personalized recommendations [14], dynamic product pricing [15] and portfolio optimisation [10]. Multi-armed bandits are algorithms that make decisions over time by balancing the exploration and exploitation of different options(arms) such that the cumulative reward over time gets maximized. Contextual bandits are an extension of the multi-armed bandit problem [13, 16, 23] where at each round the decision maker has access to some context related to the arms before deciding which arm to chose. These have emerged as a powerful approach in recent years, outperforming traditional recommender systems by effectively addressing the challenge of cold start cases. One notable algorithm in this context is LinUCB, which was introduced by [14]. It assumes a linear payoff model, where the expected payoff or reward of an action is a linear function of the associated context features. LinUCB has also shown good empirical performance in [12] even when the problem was not linear. To obtain rankings from traditional MAB, we assume that each widget is regarded as an independent arm and use its score for ranking. However, this assumption might not hold in practice, as widgets could have interdependencies. Another approach could involve creating individual Multi-Armed Bandits (MABs) for each ranking position. In this method, we would treat each widget as an arm within the MAB. The MABs' outcomes would be influenced by the previously selected arms, as displaying the same widget multiple times would not be allowed, similar to the principles of the

Ranked Bandit Algorithm (RBA) [17]. Nevertheless, this approach also falls short of adequately exploring the interdependencies among widgets and requires managing multiple algorithms simultaneously. In our approach, we have represented each widget with an embedding following the suggestion mentioned in [19] to capture interdependencies among widgets. Recently, there have been notable studies exploring various approaches to address ranking problems, Kangas et al. investigated the use of Evolutionary algorithms to model combinatorial bandits to tackle such problems [11]. In another research, the integration of Deep Neural Networks with LinUCB was explored to get a better representation of context features [20]. However, they did not delve into the analysis of diversity in their results.

3 Methodology

Let W be the set of all possible candidate widgets, and let l be the total number of slots to be shown on the page. Here, the cardinality of $W \gg l$. The content inside each widget is owned by different teams and ranking within these widgets is not controlled by this algorithm. Two prominent challenges that we faced in ranking widgets are, First, widgets are dynamic in nature. Widgets are constantly created and removed on the platform in an ongoing manner. We also observed that customer preferences and their interaction with widgets change over time. Conventional recommendation systems trained continuously tend to recommend widgets that closely resemble those with which the user has previously engaged. This leads to no exploration and only exploitation using users' preferences. To mitigate these challenges, we formulate the problem of ranking widgets as a Multi-arm Bandit (MAB) problem.

3.1 Contextual Bandit Framework

Specifically, within MAB, we use contextual bandits, where context contains both the user's and the widget's features. The User context, denoted as **u**, comprises a d-dimensional user vector $u_v \epsilon R^d$ which captures the user preference on the platform, including user browsing, add to card (ATC), and purchase history. Different time ranges are used while generating embeddings to capture users' short-term and long-term preferences. To construct **u** we draw inspiration from prod2vec [22] which employs a similar approach to generate product representation. Additionally, we leverage a k-dimensional user representation $u_{svd} \epsilon R^k$ learned through the Singular Value Decomposition (SVD) decomposition of the matrix constructed using user and widget interactions. Similarly, Widget context denoted as **w**, encompasses a m-dimensional widget vector $w_v \epsilon R^m$. This vector is created using features derived from widget meta-information and widget attributes. Furthermore, we leverage SVD decomposition to learn a k-dimensional representation of each widget $w_{svd} \epsilon R^k$. Apart from user and widget features, we have also taken other contextual features

Table 1 Notation used in this paper

Notation	Explanation	Notation	Explanation
W	Set of all possible widgets	l	Total number of slots
$\mathbf{u}$	User Context	$\mathbf{w}$	Widget context
S	item-item similarity matrix	C	Set of all customers
$\mathbb{D}_t$	Batch feedback	α	Exploration parameter
γ	forgetting factor in LinUCB	θ	Kernel parameter of DPP to control the relevance and diversity

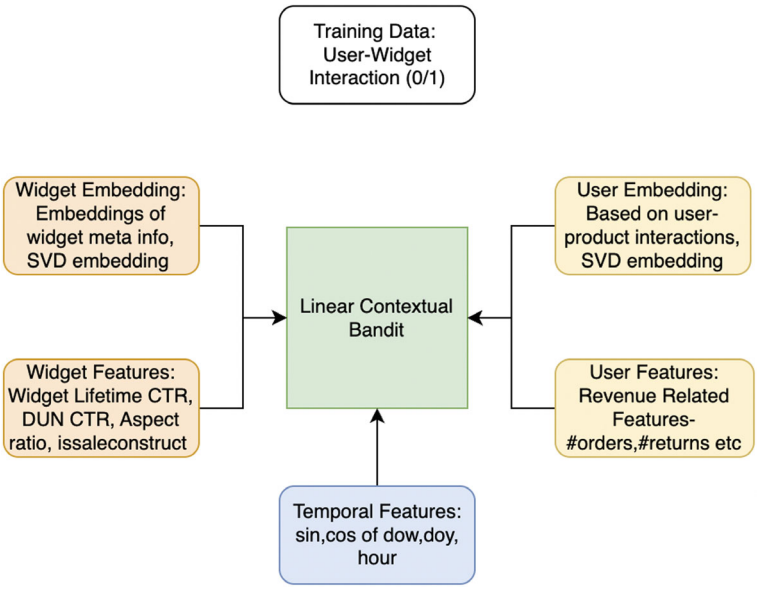


Fig. 2 Our bandit model in detail

such as 'hour of the day', 'day of the week'(dow), 'day of the year'(doy), and so on. User revenue-related features are also incorporated into the model for example 'avg no of sessions per week', 'avg bucket size' etc. We combine **u**, **w**, and other features to form a single z-dimensional context vector $\mathbf{x} \epsilon R^z$ (Table 1 and Fig. 2).

In this paper, we have explored Linear contextual bandits particularly the LinUCB and Linear Thompson Sampling (LinTS) [2]. In practice, due to resource and run-time constraints, user-widget interactions are not processed immediately, and model parameters are updated using batch feedback. This means that instead of updating the model after each individual interaction, we collect interactions over a certain period, and then update the model parameters using this aggregated feedback. Both LinUCB and LinTS possess similar representation capabilities. While LinUCB deterministically selects arms until new feedback is incorporated, LinTS adopts a stochastic approach by sampling from a posterior distribution, allowing it to randomize over actions even in the absence of updated rewards [3]. This difference

Table 2 Impact of exploration coefficient α on diversity metrics

Model	ILAD	ILMD
LinUCB ($\alpha = 0.01$)	0.48024	0.04021
LinUCB ($\alpha = 0.05$)	0.48103	0.03926
LinUCB ($\alpha = 0.25$)	0.48079	0.03942
LinTS ($\alpha = 0.01$)	0.48118	0.03959
LinTS ($\alpha = 0.05$)	0.48733	0.04127
LinTS ($\alpha = 0.25$)	0.50219	0.04185

in exploration-exploitation strategies between LinUCB and LinTS forms the basis for their comparison in this study.

Now once we have the context vector in place, CB works as follows:

- In any round t, let K_t be the number of widgets in play and we have contextual features for each widget $x_{t,i}$.
- The algorithm makes a recommendation of the current round t, based on the parameters learned using contexts x_i and rewards r_i for actions a_i $i \epsilon [1, 2, ..., t-1]$ in previous rounds and the current context x_t.
- The algorithm receives the rewards of action a_t, which are based on clicks and impressions.

In our specific setting, where we represent each widget with embedding, widgets that share similar attributes might possess highly similar representations. As a result, the bandit algorithm could end up selecting similar widgets, leading to reduced diversity in the chosen widgets. This issue arises due to the linear bandit's focus on optimizing the expected reward without explicitly considering diversity as an optimization criterion. Consequently, the algorithm may tend to exploit a narrow subset of widgets that have demonstrated favorable performance and widgets having similar representations, limiting its exploration of diverse alternatives. In Table 2 we have demonstrated the impact of exploration coefficient α on diversity metrics, as detailed in Sect. 4.2.1. The table showcases the variations in these metrics corresponding to different values of α. To compute these metrics we have taken the top 10 recommendations generated by the model. Although α plays a crucial role in the exploration-exploitation trade-off, our results indicate that diversity metrics exhibit relative stability across different values of this coefficient.

Note that the reward estimation function is the same for all the arms. This allows us to effectively handle the dynamic number of arms.

3.2 Incorporating Diversity Using DPP

Focusing solely on high-relevance items might result in a repetitive selection of similar products, disregarding the diverse preferences and interests of individual users. The lack of content diversity can hinder the exploration of potential appealing

options, thus diminishing the overall engagement and satisfaction of customers during their shopping journey. In this paper, we proposed a two-stage ranking model for widgets ranking. In the first stage, we leverage a Contextual Bandit model to generate relevance scores for the widgets. Subsequently, we employ the Determinantal Point Process (DPP) to re-rank the widgets based on their relevance scores and widget features. DPP balances the relevance of widgets and their similarities. Our approach is outlined in the Algorithm 1, which is used to rank widgets. The observed rewards corresponding to the actions are logged and used to retrain the model using batch feedback. To account for changes in the environment [8] and to give more importance to recent data samples, we incorporate a forgetting factor $\gamma \epsilon [0, 1]$ during model parameter update. Algorithm 2 is adopted from [4] which presents a fast implementation of the greedy MAP inference algorithm for DPP where they have defined diversity in the feature space of the entire subset unlike techniques based on pairwise dissimilarities. We followed a similar approach as highlighted in the paper to create the kernel matrix using $p_{t,1:K_t}$ and $X_{t,1:K_t}$. DPP selects a subset of items based on stopping criteria so $Y_t \subseteq K_t$. For the final ranking, we append the remaining widgets based on the scores obtained from the CB algorithm.

Algorithm 1: LinUCB with diversity using DPP

Input: Number of Rounds T, Number of widgets eligible per round K_t, Dimension of feature vectors z, exploration parameter α

Output: Recommended ordered widgets

Initialize: Set $\alpha > 0$, $A \leftarrow I_z$, $b \leftarrow 0_z$;

for $t = 1, 2, \ldots, T$ **do**

  $\phi_t \leftarrow A_t^{-1} b_t$;

  $\mathbb{D}_t = \emptyset$;

  Given a user we observe K_t features, $x_{t,1}, x_{t,2}, \ldots x_{t,K_t} \epsilon \mathbb{R}^z$

  for $k = 1$ **to** K_t **do**

    Calculate the upper confidence bound (UCB): $p_{t,k} \leftarrow \phi_t^T x_{t,k} + \alpha \sqrt{x_{t,k}^T A_t^{-1} x_{t,k}}$;

  end

  Ranked widgets $Y_t \leftarrow DPP(p_{t,1:K_t}, X_{t,1:K_t})$ recommend widgets Y_t, observe rewards R_t;

  $\mathbb{D}_t = \mathbb{D}_t \cup (X_t, Y_t, R_t)$

  Update the model using batch feedback $\mathbb{D}_t$

end

4 Experiments

In this section, we evaluate the performance of our framework in both offline and online evaluation. Here we have used diversity and ranking metrics to evaluate it quantitatively. We have also demonstrated the effectiveness of our framework through the A/B test.

Algorithm 2: DPP

Input: Positive semi-definite kernel matrix $L \in \mathbb{R}^{K_t \times K_t}$, *stopping criteria*
Output: A subset of items Y_t

Initialize: $c_i = []$, $d_i^2 = \boldsymbol{L}_{i,i}$, $j = \text{argmax}_{i \in K_t} \log(d_i^2)$, Set $Y_t = \{j\}$;
while *stopping criteria not satisfied* **do**
  for $i \in K_t \setminus Y_t$ **do**
    $e_i = (\mathbf{L}_{j,i} - (\mathbf{c}_j \cdot \mathbf{c}_i))/d_j$;
    $\mathbf{c}_i = [\mathbf{c}_i, e_i]$; $d_i^2 = d_i^2 - e_i^2$;
  end
  $j = \text{argmax}_{i \in K_t \setminus Y_t} \log(d_i^2)$;
  $Y_t = Y_t \cup \{j\}$;
end

4.1 Offline Results

In an industrial setup, off-policy evaluation is challenging. Existing approaches require the data to be randomly logged or the current policy in production to be similar to the test policy [7, 9]. However, such assumptions rarely hold in practical scenarios, as the data received in an industrial setup tends to be purely exploitative and generated by statically positioned widgets for all the customers. We evaluate our algorithm using the traditional offline method where we combine the user and widget contexts and feed it into our framework which generates a ranked order of widgets. We have taken the user widget interaction data for a couple of days to train the model.

4.1.1 Metrics

We have used Normalized discounted cumulative gain (nDCG), Intra-list average distance (ILAD) [24], and Intra-list minimal distance (ILMD) as performance metrics. They are defined as:

The nDCG formula is given by:

$$\text{nDCG} = \frac{\text{DCG}}{\text{IDCG}},$$

where DCG (Discounted Cumulative Gain) is defined as:

$$\text{DCG} = \sum_{i=1}^{n} \frac{2^{rel_i} - 1}{\log_2(i+1)},$$

and IDCG (Ideal Discounted Cumulative Gain) is the DCG score obtained with the ideal ranking of the items.

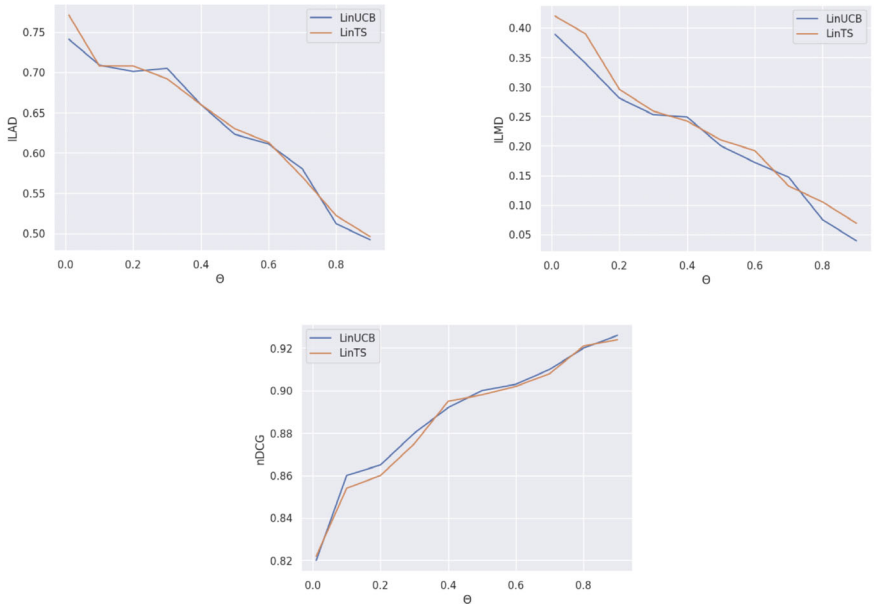


Fig. 3 Influence of θ on relevance (nDCG) and diversity metrics (ILAD, ILMD)

The Diversity metrics are defined as:

$$\text{ILAD} = \underset{c \epsilon C}{mean} \underset{i,j \epsilon R_c, i \neq j}{mean} (1 - S_{i,j})$$

$$\text{ILMD} = \underset{c \epsilon C}{mean} \underset{i,j \epsilon R_c, i \neq j}{min} (1 - S_{i,j})$$

As we can see from Fig. 3 we captured the impact of varying the trade-off parameter $\theta \epsilon [0, 1]$ of DPP [4] w.r.t ILAD and ILMD. Both metrics demonstrate an almost monotonic decreasing behavior with respect to θ, while nDCG exhibits an increasing pattern. Here the higher value of θ implies more relevance. We identify a promising range, $0.6 \leq \theta \leq 0.8$, where a desirable trade-off between relevance and diversity can be achieved in our widget ranking case.

4.2 Online Results

We warm-started the model using the last 7 days of data before launching the A/B test. We evaluated the XGBoost model alongside linUCB in a production environment. Our results demonstrated an improvement of **0.24%** in Home page clicks per user over the XGBoost model with a high statistical significance which is a decent

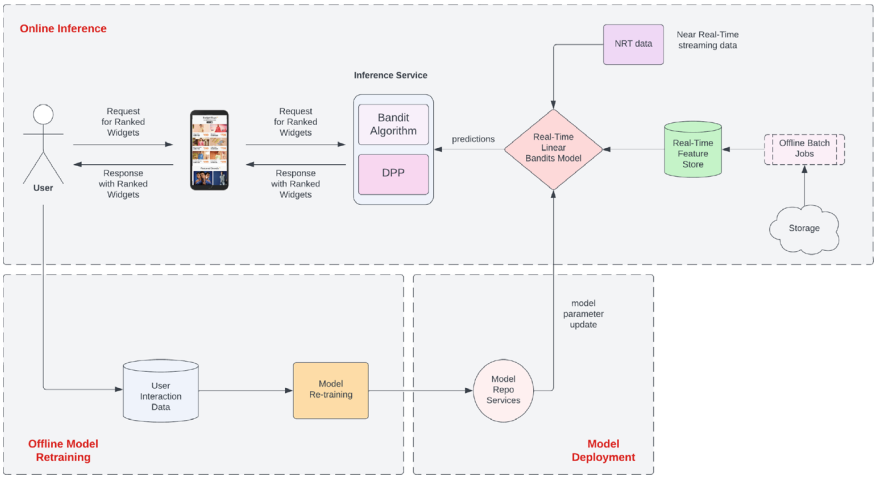


Fig. 4 Production system architecture. Our production system consists of (1) online inference that coordinates the frontend request and generates ranked widgets by executing Inference Service, (2) offline model retraining in batch mode after every time 't', and (3) model versioning and deployment

gain given our scale. All other guardrail metrics such as RPU (Revenue per user), user conversion, etc. also exhibited positive outcomes. Figure 4 shows the overall production system architecture.

5 Conclusion and Discussion

In this paper, we modeled the personalized widgets ranking as a contextual multi-armed bandit problem focusing on improving the overall shopping experience of the customers. Therein, we addressed the challenges faced in ranking widgets, we highlighted the benefits of our system, notably modeling the widgets with features so as to cater to different types of widgets in a single ranking purview and integrating it with DPP to achieve diversity in ranking. We have shown offline results of our framework where we compared LinUCB and LinTS. This framework can easily be extended to any bandit algorithm and it can be served not just on the homepage but to any other store pages. Future work can include a multi-objective optimization or defining a reward in such a way that it represents an aggregate of different objectives, we believe this can be a good research direction.

References

1. Abbasi-Yadkori Y, Pál D, Szepesvári C (2011) Improved algorithms for linear stochastic bandits. In: Advances in neural information processing systems, p 24
2. Agrawal S, Goyal N (2013) Thompson sampling for contextual bandits with linear payoffs. In: Dasgupta S, McAllester D (eds) Proceedings of the 30th international conference on machine learning. In: Proceedings of machine learning research, vol 28. Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR, pp 127–135
3. Bendada W, Salha G, Bontempelli T (2020) Carousel personalization in music streaming apps with contextual bandits. In: Proceedings of the 14th ACM conference on recommender systems, pp 420–425
4. Chen L, Zhang G, Zhou E (2018) Fast greedy map inference for determinantal point process to improve recommendation diversity. In: Advances in neural information processing systems, p 31
5. Cheng H-T, Koc L, Harmsen J, Shaked T, Chandra T, Aradhye H, Anderson G, Corrado G, Chai W, Ispir M, Anil R, Haque Z, Hong L, Jain V, Liu X, Shah H (2016) Wide & deep learning for recommender systems. CoRR, arXiv:1606.07792
6. Chu W, Li L, Reyzin L, Schapire R (2011) Contextual bandits with linear payoff functions. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, pp 208–214
7. Gilotte A, Calauzènes C, Nedelec T, Abraham A, Dollé S (2018) Offline a/b testing for recommender systems. In: Proceedings of the eleventh ACM international conference on web search and data mining, pp 198–206
8. Graepel T, Candela JQ, Borchert T, Herbrich R (2010) Web-scale Bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. Omnipress
9. Guo D, Ktena SI, Myana PK, Huszar F, Shi W, Tejani A, Kneier M, Das S (2020) Deep Bayesian bandits: exploring in online personalized recommendations. In: Proceedings of the 14th ACM conference on recommender systems, pp 456–461
10. Huo X, Feng F (2017) Risk-aware multi-armed bandit problem with application to portfolio selection. R Soc Open Sci 4(11):171377
11. Kangas I, Schwoerer M, Bernardi L (2022) Scalable user interface optimization using combinatorial bandits. In: Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval, pp 3375–3379
12. Krishnamurthy A, Agarwal A, Dudík M (2015) Efficient contextual semi-bandit learning. CoRR, arXiv:1502.05890
13. Langford J, Zhang T (2007) The epoch-greedy algorithm for multi-armed bandits with side information. In: Platt J, Koller D, Singer Y, Roweis S (eds) Advances in neural information processing systems, vol 20. Curran Associates, Inc.
14. Li L, Chu W, Langford J, Schapire RE (2010) A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the 19th international conference on World wide web, pp 661–670
15. Misra K, Schwartz EM, Abernethy J (2019) Dynamic online pricing with incomplete information using multiarmed bandit experiments. Mark Sci 38(2):226–252
16. Pandey S, Agarwal D, Chakrabarti D, Josifovski V (2007) Bandits for taxonomies: a model-based approach. In: Proceedings of the 2007 SIAM international conference on data mining. SIAM, pp 216–227
17. Radlinski F, Kleinberg R, Joachims T (2008) Learning diverse rankings with multi-armed bandits. In: Proceedings of the 25th international conference on machine learning, pp 784–791
18. Rendle S (2010) Factorization machines. In: 2010 IEEE international conference on data mining, pp 995–1000
19. Saito Y, Joachims T (2022) Off-policy evaluation for large action spaces via embeddings. arXiv:2202.06317

20. Shi Q, Xiao F, Pickard D, Chen I, Chen L (2023) Deep neural network with linucb: a contextual bandit approach for personalized recommendation. In: Companion proceedings of the ACM web conference 2023, WWW '23 Companion, New York, NY, USA, 2023. Association for Computing Machinery, pp 778–782
21. Thompson WR (1933) On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika 25(3–4):285–294
22. Vasile F, Smirnova E, Conneau A (2016) Meta-prod2vec: product embeddings using side-information for recommendation. In: Proceedings of the 10th ACM conference on recommender systems, pp 225–232
23. Wang C-C, Kulkarni SR, Vincent Poor H (2005) Bandit problems with side observations. IEEE Trans Autom Control 50(3):338–355
24. Zhang M, Hurley N (2008) Avoiding monotony: improving the diversity of recommendation lists. In: Proceedings of the 2008 ACM conference on recommender systems, pp 123–130

Correction to: UNICON: A Unified Framework for Behavior-Based Consumer Segmentation in E-Commerce

Manuel Dibak, Vladimir Vlasov, Nour Karessli, Darya Dedik, Egor Malykh, Jacek Wasilewski, Ton Torres, and Ana Peleteiro Ramallo

Correction to:
Chapter 4 in: N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_4

In the original version of Chapter 4, the author Yashar Deldjoo was incorrectly listed as a contributor. This has now been rectified and the correction has been updated.

The updated version of this chapter can be found at
https://doi.org/10.1007/978-3-031-76878-1_4

N. Dokoohaki et al. (eds.), *Revolutionizing Fashion and Retail*, Lecture Notes in Electrical Engineering 1299, https://doi.org/10.1007/978-3-031-76878-1_8