CIS 522 – Final Project – Technical Report

Fully Connected Model

April 2022

Team Members:

- Chen Fan; cfan3; Email: `cfan3@seas.upenn.edu`
- Yichun Cao; ycao4; Email: `ycao4@seas.upenn.edu`
- Yixuan Meng; yixuanm; Email: `yixuanm@seas.upenn.edu`

Abstract

The problem of information overload is growing explosively especially in the digital age, and research on recommender algorithms has gained much attention since it provide users with personalized content and timely access to items of interest. Our project aims at developing product recommender systems based on the transactions history and meta data of customers and products from *H&M* Group. We implement User-to-user Collaborative Filtering and Content-based Filtering as non-deep learning baselines. And build a DNN as a candidate generation model. We argue that transfer learning helps to encode more information using pre-trained model. Further, a session-based recommendations is implemented. We compare and evaluate the models using Mean Average Precision for top 12 recommendations.

1 Introduction

On the Internet, where the number of choices is overwhelming, there is need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of information overload [IFO15]. Recommender algorithms solve such problem by hunting over tremendous volume of dynamically generated ocean of information to provide users with customized options that greatly reduce the redundant cost of browsing through endless web pages. On merchants' side, advances in recommender algorithms provides precise indicator for market trends and pricing on merchants' side. Additionally, the topic has always been a hit in the academic field. Numerous competitions on recommender systems have been held due to pursuit for personalized AI-based advertisement by all kinds of companies, such as traditional retails like *H&M* Group, tech companies like Google, Facebook, and streaming service companies like Netflix and YouTube.

This paper explores three different methods with distinct characteristics and prediction techniques in recommendation systems using massive meta data provided by $H\&M$ Group. Specifically, given all kinds of data on customers, products, and purchase history, we aim for designing, implementing and improving recommender algorithms and models that yield 12 potential recommends for each customers in the given dataset. By such design, we can experiment and compare different classical strategies with modified algorithms with a specific example. As the current researches regarding recommender systems mostly focuses on one branch, which Collaborative Filter or Session-based recommendation in depth. This project stands out from other researches regarding this topic that it explores and compares three models altogether including Machine Learning and Deep Learning models, and aims at figuring out an algorithm that situate the data better.

For the purpose of making high-quality and personalized recommendations, in this project we have investigated popular recommender algorithms relying on long-term user profile such as Collective Filtering. To evaluate our models, we use an evaluation metrics called Mean Average Precision for the 12 recommendations generated by the models, namely MAP@12 score.

The key contributions of this project are summarized as follows:

- For non deep learning models, We implement User-to-user Collaborative Filtering (UUCF) and Content-based Filleting (CBF) with Principal Component Analysis (PCA), achieving 0.0025 MAP@12 score.
- We research on candidate generation models for recommenders and evoted to implement Deep Neural Network (DNN) as a candidate generation model, which achieve 0.0158 MAP@12 score.
- We improve the performance of the DNN model by transfer learning method with pre-trained BERT (Bidirectional Encoder Representations from Transformers) model and reach 0.0179 MAP@12 score.
- We apply session-based recommendations using GRU unit and improved MAP@12 score to 0.0210.

With the above introduced models, this paper combines the result, analyzes the performance of each model and yields a general analysis for which model is a better solution for this problem.

2 Related Work

Isinkaye [IFO15] talked about the history and multiple models focusing on filtering techniques in recommender system, which includes Content-based Filtering (CBF) and Collaborative Filtering (CF). As the author discussed, in Content-based model, we match user features and item features and recommend product with highest relation to users. This method does not require much information from other users than the target user, therefore it's more adaptable to changes

than other models. Among all those CF, we choose a user-based CF, UUCF. Different from CBF, CF highly relies on other users' information which uses similarities of users as benchmark to make recommendations. Isinkaye also pointed out that CF has some critical issues such as data sparseness and cold-start problem. Therefore, in our CF approach, we filter out the users made less than two purchases in recent six months.

Besides the traditional filtering techniques, there are lots of related works on Deep Learning techniques for recommendation. The author of [CAS16] proposes a sophisticated industrial recommendation systems for Youtube videos. The system comprises of two neural networks, a candidate generation model and candidate ranking model. The former give a small subset of videos as recommendations from huge amount of Youtube videos based on users watch history. The ranking model then choose top N items by computing a scores for each video according to desired features of video and user. The attention mechanism[Vas+17] has the ability to decide which part of input texts is worth to focus on. Transformer models with attention mechanism such as BERT[Dev+18], RoBERTa[Liu+19], XLNET[Yan+19] are pre-trained on huge corpus and are proved to be effective language models to use in previous studies.

As the previous main algorithms are based on the long-term profiles of customers, in many real-world applications the records may not be available and another algorithm is designed called session-base algorithm. As Ludewig and Jannach introduced in [LJ18], session-based recommendations have to be made solely based on the observed behavior of a user during an ongoing session, aiming to predict the user's immediate next actions. The paper [Hid+15] introduced the most recent approaches based on recurrent neural networks of GRU4REC, which is also the model we applied in this project. This approach considers practical aspects of the task and introduces several modifications to classic RNNs, and specific to this project this method and model is easier to transplant into the given dataset. Rendle discussed session-based recommender using factorized Markov model which combine the two most popular approaches based on matrix factorization (MF) and Markov chains (MC) in [RFS10]. Aside from the mentioned two recent approaches, a simpler methods based on nearest neighbor schemes is expanded and evaluated in paper [Lud+21].

3 Dataset and Features

3.1 Dataset

We use the H&M Personalized Fashion Recommendations dataset[1] in our work. This dataset released by the H&M Group, is collected from real purchase data and contains three files, provide meta data of customers, meta data of products, and purchase history.

[1] https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations/data

The customer table has information of 1,371,980 customers with unique customer ID. It collected age of customers, their club member status (active or not), frequency of receiving fashion news, etc. We visualize the distribution of customers' ages and whether they are active as shown in Figure 1. The number of active and inactive customers are quite balanced.

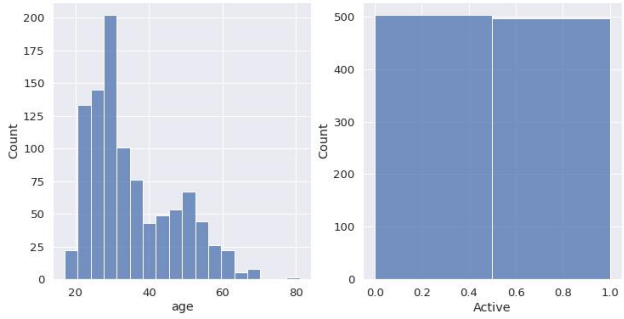

Figure 1: Distribution of customers' ages and active

The product table has available meta data spans from product name, color, department, to detailed descriptions. There are 105,542 different articles and the article ID is unique for each product. An example of product data is shown in Table 1. There are multiple columns that representing the same information, like 'colour_group_code' and 'colour_group_name', we use the text columns in our work.

article_id	0108775015
product_code	0108775
prod_name	Strap top
product_type_name	Vest top
product_group_name	Garment Upper body
graphical_appearance_name	Solid
colour_group_code	09
colour_group_name	Black
detail_desc	Jersey top with narrow shoulder straps.

Table 1: Example of product data

We visulize the word cloud for product descriptions, as shown in Figure 2. The word cloud shows that upper fits have high sales, and dark black is a

popular color.

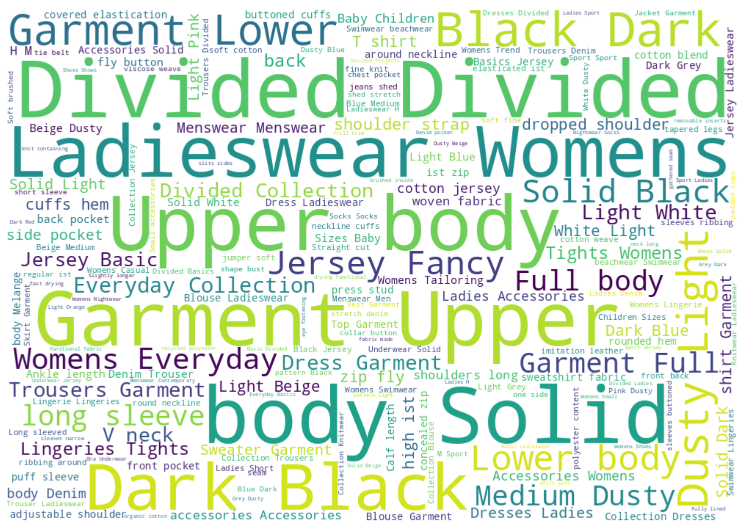

Figure 2: Word cloud for product description

The transaction table records previous purchases, it consists of the purchases each customer for each date, as well as additional information. As shown in Table 2, the table provides the date of the purchase, the customer and article ID, and the price. The dataset is collected from September 2018 to September 2020.

t_dat	2018-09-20
customer_id	000058a12d5b43e67d225668fa1f8d618c13dc232df0cad8ffe7ad4a1091e318
article_id	0663713001
price	0.050830508474576264

Table 2: Example of transaction data

3.2 Features

Due to the large scale of this dataset, we sampled most active customers from recent data to reduce the data size. Specifically, we selected 2000 customers with the most purchase history from the last month of the time span, which is from August 22, 2020 to September 22, 2020. We further filtered the product table to only contain products that were bought by those customers and filtered the transaction table to only include purchase history of the 2000 customers. There are 43,121 purchase history left, and 9,925 unique articles after applying this filter.

Based on the purchase history and meta data of products and customers, we create the following features.

1. Word embedding
   Using text from the product table, we generate GloVe embedding for each product. We put together the name, type name, color name, detailed descriptions of products and then train the 300-dimensional embedding on the corpus. The size of the embedding is padded to 70 for each product, and the resulted vocabulary size is 15,743.

   In the transfer learning method, we use the BERT tokenizer to generate word embeddings. The padded length is still set to 70. The tokenizer is pre-trained on large corpus and we assume that it carries relevant information when encoding product data.

2. Customer context
   We use four features of customers in total. The 'FN' column in the customer table represents whether the customer receive fashion news, we process it to be binary and use it as a feature. The 'Active' column is also a binary variable. We process the 'club member status' column so that it is none, pre-active or active. We also use the age of the customers.

3. Purchase history
   According to the transaction table, we find a list of product that each customer bought. We map the product ID to their embedding and create corresponding list of word embedding of the products. Since customers purchased different number of products before, we take the average of the embedding to make their length consistent.

   To generate true label for training and evaluation, we build a binary matrix of shape (number of customers, number of products), so that each row represents a customer and each column represents a product. The element of the matrix is 1 if the customer purchased the product before, and is 0 otherwise.

4 Methodology

4.1 Filtering Techniques

Filtering Technique is a commonly used in Recommender system, which is a good fit for our project goal. In this project, we did two types of Filtering Techniques, the User-to-user Collaborative Filtering (UUCF) and Content-based Filtering (CBF) using PCA.

- UUCF: In this method, we make the information for each user as vector, u, then we find similar users for u by calculating overlaps in purchased items, which we called neighborhood of u. Among the neighborhood of u, we find top 6 most popular items ranked by the number of purchases made in the neighborhood.

- CBF with PCA: The purpose of CBF is to calculate the similarity of an item and items a user purchased before by constructing a user-feature matrix and item-feature matrix. We use one hot encoding to change the categorical data, such as "product group name" into one-hot vector for both user-feature matrix and item-feature matrix. Then we normalize the user-feature matrix, and use the dot product of normalized user-feature matrix and item-feature matrix as our similarity score matrix. For any user we want to predict, we just find 6 items with highest scores. In this method, we use PCA to reduce the dimensionalities of data and increase interpretability.

4.2 Deep Neural Network

To make full use of product descriptions and customer history, we build a deep neural network (DNN)[CAS16] as a recommender system.

The recommender system has a massive user base and involves a large volume of products, and it is likely to be dynamic and has new purchase history as inputs in the real world, the deep neural network is supposed to be an appropriate setting for this problem.

Our network is referred as candidate generation model. It is able to take the purchase history of customers as inputs, and generate a dozen of products as recommendation from H&M's plenty of products.

We used the features from section 3.2, including the GloVe embedding, demographics of customers, and purchase history to decide the similarities between users and products. In Figure 3, we show the system overview for the candidate generation model.

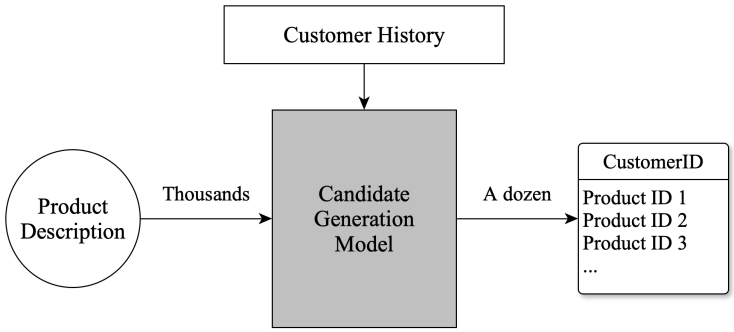

Figure 3: System overview for the candidate generation model

To explain further, the model uses a linear layer to extract customer features first and concatenates the extracted feature with the vectors representing the purchase history. We then feed the customer history and contexts into two more linear layers with dropout, then multiply the matrix of embedding vectors encoded the products description. The dropout rate is set to 0.5 and we use a

sigmoid function to get the final output. The top 12 recommendations for each customer are found by softmax. The transaction data is split into training and validation set, the validation set consists of the most recent 20% data. During training, the batch size is set to 4, and the learning rate is 0.0005. We use the Adam optimizer and use the binary cross entropy loss as the criterion. The model is trained for 10 epochs.

4.3 Transfer Learning

BERT(Bidirectional Encoder Representations from Transformers) is a pre-trained language model based on a multi-layer bidirectional transformer encoder. Since it is pre-trained with a large corpus, fine-tuning it for our task is efficient. We used a base cased BERT model from the Huggingface, and use the tokenizer to generate a masked input.

Similar to what we did for DNN, we first uses a linear layer to extract customer features. Bu instead of taking average GloVe embedding of all purchased items of a customer, we put together all product descriptions of the list of products, and use BERT tokenizer to generate the token embeddings, position embeddings, and segment embeddings. We feed these emdeddings into the pre-trained model to extract features for the purchase history. Again we concatenate the customer history and context, and add linear layers after the pre-trained model. We freeze the parameter of BERT, and fine-tune on the fully connected layers.

To compare the effect of using the pre-trained model, we use the same setting (batch size, optimizer, criterion) as the DNN model only with linear layers. For this transfer learning setting, we train the model for 5 epochs.

4.4 Session-based Recommendations

In a real-life setting where long-term user preferences may not be available, recommendations have to be made solely based on users' current browsing session. These results, proven in previous research, may outperform today's complex recommendation methods relying on long-term yet static user profiles in terms of prediction accuracy, since the referencing data is more updated. Although the dataset is not specifically built for such pure short-term based algorithm since ton of information is left untouched, it is worth experimenting and comparing with the two proposed methods.

Some adjustments are needed to accommodate this algorithm. In a regular session-based recommendation, users' clicks or longer dwell time are used as initial inputs of the RNN, and each consecutive click will produce recommendations based on all previous activities. In order to mimic such setting, we would take the purchase history within a relatively short time frame as the consecutive click events and slightly modify the question into make 12 predictions of last item based on previous "clicks". Additionally, only data from customers who purchased over 40 items is taken into account, because the click-stream dataset

is typically quite large and setting some limit better enables the consecutive click imitation.

The model used is a GRU-based recurrent neural network, which is proposed by [Hid+15]. The input of the network is the actual state of the session while the output is the item of the next event in the session, and the general layout of the model is summarized in figure 4. 1-of-N encoding is used in the embedding layer which ensures that input vector's length equals to the number of items as 64 with padding for insufficiency and truncating for surplus. Hyper-parameter tuning with random search is used for finding relatively optimal learning rate and dropout percentage. In the finalized version, learning rate is 0.001 and dropout rate is 0.3. The Gated Recurrent Unit (GRU) is a more elaborate model of an RNN unit that aims at dealing with the vanishing gradient problem [Hid+15]. It besides the additional feedforward linear layer is added such that the model essentially learn when and by how much to update the hidden state of the unit.

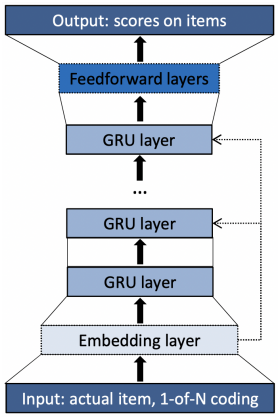

Figure 4: Structure summary for GRU-based RNN model

5 Results

We evaluate our models according to the Mean Average Precision @ 12 (MAP@12):

$$MAP@12 = \frac{1}{U}\sum_{u=1}^{U}\frac{1}{min(m,12)}\sum_{k=1}^{min(n,12)}P(k)\times rel(k) \tag{1}$$

where U is the number of customers, $P(k)$ is the precision at cutoff k, n is the number predictions per customer, m is the number of ground truth values per customer, and $rel(k)$ is an indicator function equaling 1 if the item at rank k is a relevant (correct) label, zero otherwise.

We choose this evaluation metric because it is independent of whether a customer made a purchase during text period or not (i.e. exclude non-purchase

customers from calculating the score). And there is no penalty for generating 12 recommendations if the customer purchase less than 12 products, which makes sure that generating 12 recommendations for each customer as we did in our work makes sense.

The full results of the MAP@12 metric is summarized in Table 3.

Method	MAP@12
User-User Collaborative Filtering	0.0028
Content-based Filtering	0.0025
Deep Neural Network	0.0158
DNN with Transfer Learning	0.0179
Session-based recommends with RNN	0.0211

Table 3: MAP@12 Results

We further analyse our results by visualizing the recommendations of our models and the real purchases of the customers during validation time period.

5.1 Collaborative Filtering Techniques

5.1.1 UUCF

In our UUCF model, we return top 12 popular products in a neighborhood of 30 similar users. The following is the recommendation for one user

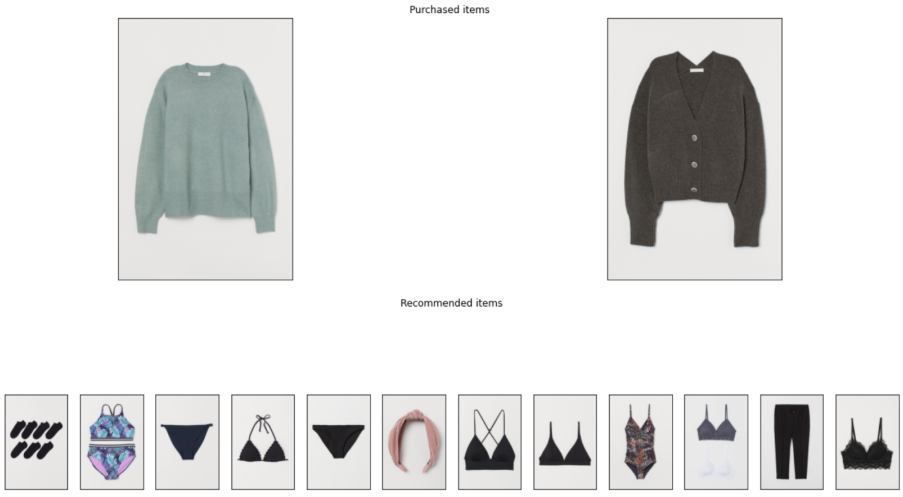

Figure 5: Recommendations by UUCF

5.1.2 CBF

In CBF, we return 12 items that have highest relation scores calculated by the dot product of item-feature matrix and user-feature matrix. The following is one example of recommendation by CBF:

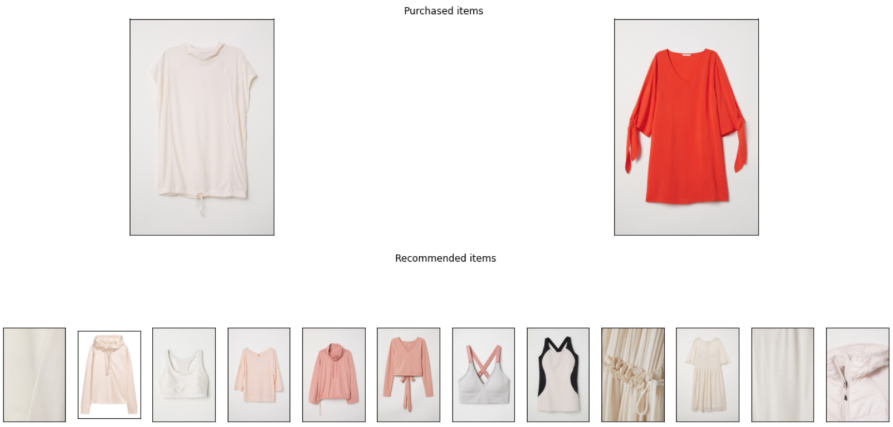

Figure 6: Recommendations by CBF

5.2 Deep Neural Network

The batch size is set to 4, and the learning rate is 0.0005. We use the Adam optimizer and use the binary cross entropy loss as the criterion. The DNN is trained for 10 epoches, the train loss and validation loss during training is shown in Figure 7.

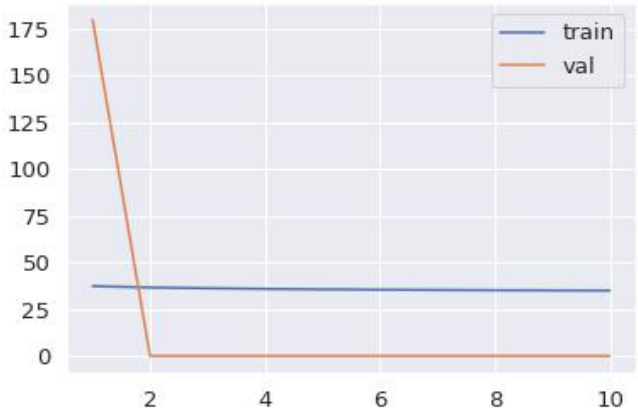

Figure 7: Train loss and validation loss of DNN

We show an example of real purchased item of a customer, and the predicted recommendations in Figure 8. According to the figure, the model successfully predicted one of products this customer purchased during validation time period. The product is highlighted in the figure.

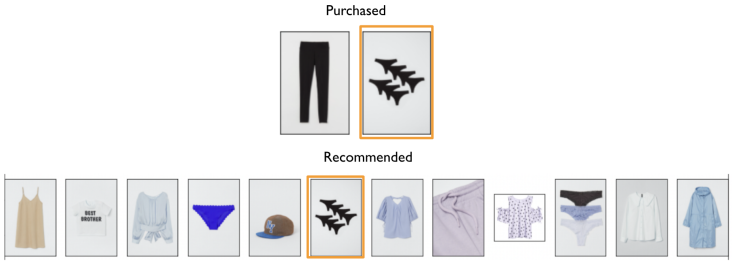

Figure 8: Actual purchased products versus predicted recommendations of DNN

5.3 Transfer Learning

The batch size is set to 4, and the learning rate is 0.0005. We use the Adam optimizer and use the binary cross entropy loss as the criterion. This transfer learning model is also trained for 10 epoches, the train loss and validation loss during training is shown in Figure 9.

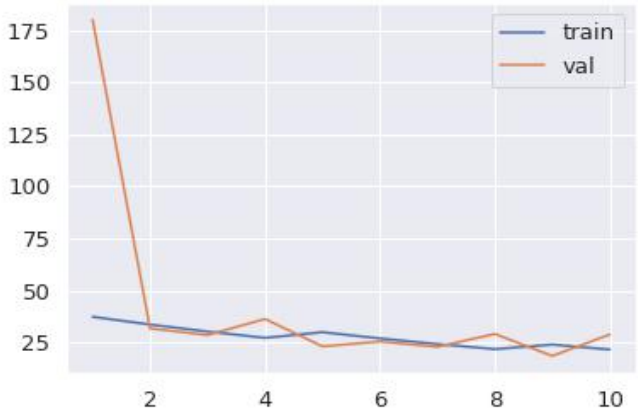

Figure 9: Train loss and validation loss of transfer learning model

We also show an example of actual purchased item and the predicted recommendations for a customer in Figure 10. According to the figure, the model not only successfully predicted one of products this customer purchased during validation time period, but also tried to recommend products that are similar to what the customer will buy. We can see that the trouser in the purchased products also appears in the recommended products, the purchased and recommended list of products both have underwear.

Figure 10: Actual purchased products versus predicted recommendations of transfer learning model

5.4 Session-based recommender with RNN

The learning rate is 0.001, and the drop out rate is 0.3. We use the optimizer and use cross entropy loss as the criterion. The model is trained for 70 epochs, the train loss is shown in Figure 11.

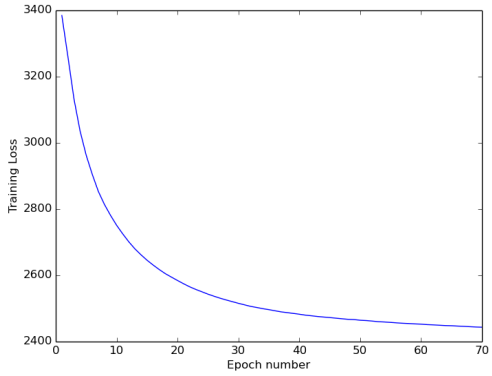

Figure 11: Train loss of session-based recommendation with RNN model

Shown below in Figure 12 is an example of one customer's purchased history in 2020 and the 12 recommendations generated by the RNN model. There are a few things to notice in the results. First is that the recommends include the user's purchase history, and this can be explained through concept of session-based recommendations to make suggestions based on click events which are presumably unpurchased yet interested items. Other than re-recommending, the model performs well in making predictions that aligns with input either in style, in color, in pattern, or in category.

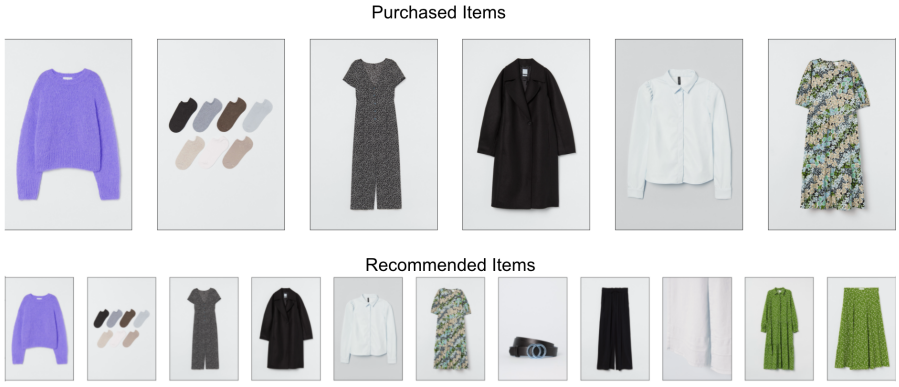

Figure 12: Actual purchased products versus predicted recommendations of session-based recommendations with RNN model

6 Discussion

In this project, we explore different methods to build a recommender system. We use User-user Collaborative Filtering and Content-based Filtering as non-deep learning baselines and proceeded to build a deep neural network. We integrate the meta data of products and customers by generating GloVe embedding and BERT embedding and concatenate the features with purchase history data. We compared the performance of the DNN model and the results of transfer learning. Moreover, in order to overcome usual absence of long-term user profile and imitate some real-life recommender algorithms, we apply session-based recommendations with RNN, which focused on temporal order of purchased products by predicting the last purchased item based on fresh knowledge within the same session.

6.1 Findings

In the Filtering techniques, the UUCF reaches 0.0028 using MAP@12, while the CBF gets 0.0025. Among these two filtering techniques, we can see that in UUCF, the model's recommendations are products unrelated to the products the user purchased before, and CBF is recommending products very similar to the ones that the user bought. Since the recommender system is simulating human behaviors and it's not likely that people will buy products they just bought recently, it's reasonable that CBF under-performs UUCF in our scenario.

UUCF is good at recommendation when there is not much content given with items given its benchmark by comparing user information. However, the cold-start and data sparsity problem still influence our final performance, and the computation grows linearly with the number of users and items. Given our current situation, we come up with the deep neural network model.

The deep neural network out-perform the User-user Collaborative Filtering because the neural network did better work on extracting features and deciding the similarity between customers and products.

However, the DNN method still depends on the long-term user profile and purchase history spans at least one year, and does not overcome the problem of the cold-start and data sparsity. Hence, we decided to experiment with the session-based recommender which only depends on short-term information. Implementation of the algorithm with GRU-based RNN achieves the highest score of 0.0211 using MAP@12. Simple as it relies solely on the user's actions in an ongoing session, the session-based recommender is trained on the most recent data and reflects a commence sense that fashion is strictly of timeliness. An additional proposed reason for the outperformance is that since half of users are inactive, the problem of data sparsity is greatly affecting the results.

6.2 Limitations and Ethical Considerations

For all the models we used, due to data sparsity we had to throw away majority of data to be able to run the model. Consequently, we have very limited infor-

mation to decide the similarity between products and customers, because most products are only purchased once by a single customer. This limitation causes the unsatisfying performance for models of Collective Filter and DNN models that rely on long-term user profile. Also, it greatly confines the potential of session-based model since it has an assumption that click events are quite large. Due to the nature of all three recommender algorithms, they tend to predict popular products which may hinder the sales of less popular products. In reality, some human intervention is probably needed for promotion of some products.

Specifically, the deep neural network only has simple linear layers. Although, this shortcoming is made up by the transfer learning extension we did, there are more other potentially improvement we did not tried, like using a convolutional layer.

One of a limitations regarding the generalization of our method is that transferring the recommender systems in this work to another field of study could be hard. Because building the system requires large size of data, and it would take a lot of effort to organize data and create the features. Besides, for the evaluation metrics we reported in our work, it is hard to expect a similar metric on a different dataset, because it is sensitive to the proportion of people and recommended items. Due to the same reason that the model is hard to transfer into other fields, misusing the model is not among the top concerns.

The ethical concerns are mostly regarding the given data. For privacy and security reasons, *H&M* Group only reveals very few features in the customer dataset. Though understandable, it leaves obstacle both in constructing models and in analyzing if the data is biased towards some group. Since the models focus mostly on resemblance between products, ethical concerns are not as explicit at this stage.

6.3 Future Research Directions

We proposed different methods to build recommender systems for H&M product recommendations. Potential future work are:

1. Model Structure
   The dataset we use also provides corresponding images for products. We believe that adding the extra information by aligning images and texts as input features would improve the quality of recommendations. Since the deep neural network uses linear layers, another direction of extending this work is to use a convolutional neural network. It is possible to split the purchase history into training windowsand shift the window over the sequence of user to get the embedding.

2. Compute Power
   We believe that there is still a big room for improving the MAP@K score given that we only use a small subset of purchase data. With more compute power, we will be able to consider a longer time span and cover more customers.

3. Domain Transfer
   The method we use can also be applied to similar problems in other domains, like recommending videos, books, bloggers, etc. That being said, as discussed in the former subsection for limitations, for different domains, it takes time to build corresponding features, and may require effort to use labels that are more aligned with specific purpose, e.g. recommending horror films instead of films would require the system focusing on horror-related video contexts.

6.4 Social impacts

The personalized recommendation helps users to take correct decisions in their online transactions, increase sales and redefine the user's web browsing experience, retain the customers, enhance their shopping experience. The more precise recommending algorithms generate a win-win situation where the merchants gain sales and customers have better shopping experiences and more suitable items. In a broader context, such algorithms or models prompt the economy.

Nowadays, the fashion trend is determined by some sort of committee, such as the famous Pantone color of the year. Such uniformity or trendiness reduces aesthetic diversity. Hence, the recommending algorithms and model serves as a tool for users to find their own styles and thus prompts the idea of being unique individual to reveal their own identity.

7 Conclusions

Not only recommending personalized fashion but the recommender system can also be applied to almost everything in our lives. In this paper, we applied three kinds of models: Filtering techniques, DNN-based, and RNN-based on the HM dataset, and discussed our findings on their strengths and challenges. Our latest model, the RNN-based one, adopts a new idea of session-based recommendation, which has not been well researched. In our results, we can see an obvious improvement in the performance of this method. These findings can serve as a road map for other researchers to improve the power of the recommender system. With more research exploring the recommender system and outputting more personalized information, the internet will continue to make life better.

8 References

References

[CAS16] Paul Covington, Jay Adams, and Emre Sargin. "Deep neural networks for youtube recommendations". In: *Proceedings of the 10th ACM conference on recommender systems*. 2016, pp. 191–198.

[Dev+18] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[Hid+15] Balázs Hidasi et al. "Session-based recommendations with recurrent neural networks". In: *arXiv preprint arXiv:1511.06939* (2015).

[IFO15] F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh. "Recommendation systems: Principles, methods and evaluation". In: *Egyptian Informatics Journal* 16.3 (2015), pp. 261–273. ISSN: 1110-8665. DOI: https://doi.org/10.1016/j.eij.2015.06.005. URL: https://www.sciencedirect.com/science/article/pii/S1110866515000341.

[Liu+19] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[LJ18] Malte Ludewig and Dietmar Jannach. "Evaluation of session-based recommendation algorithms". In: *User Modeling and User-Adapted Interaction* 28.4 (Dec. 2018), pp. 331–390. ISSN: 1573-1391. DOI: 10.1007/s11257-018-9209-6. URL: https://doi.org/10.1007/s11257-018-9209-6.

[Lud+21] Malte Ludewig et al. "Empirical analysis of session-based recommendation algorithms". In: *User Modeling and User-Adapted Interaction* 31 (Mar. 2021), pp. 149–181. ISSN: 1573-1391. DOI: 10.1007/s11257-020-09277-1. URL: https://doi.org/10.1007/s11257-020-09277-1.

[RFS10] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. "Factorizing personalized Markov chains for next-basket recommendation". en. In: *Proceedings of the 19th international conference on World wide web - WWW '10*. Raleigh, North Carolina, USA: ACM Press, 2010, p. 811. ISBN: 978-1-60558-799-8. DOI: 10.1145/1772690.1772773. URL: http://portal.acm.org/citation.cfm?doid=1772690.1772773 (visited on 04/27/2022).

[Vas+17] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[Yan+19] Zhilin Yang et al. "Xlnet: Generalized autoregressive pretraining for language understanding". In: *Advances in neural information processing systems* 32 (2019).