Recommender Systems Challenge 2022

JULIAN BACKÉ (01304563), PAWEL ZAGORSKI (12038923), DANIEL KANCSAR (12102459), ARTJOLA GANELLARI (12046001), ERALDA RUCI (12038076)

The course project consists of working on the 'RecSys Challenge 2022', a competition organized by Dressipi – a leading company for reccommender systems in the field of fashion. This year, the competition task focused on recommending fashion items given user sessions, purchase data, and product characteristics. The approaches used to fulfill this task are two collaborative filtering techniques item-item and user-user, as well as a content-based approach and neural network model containing a long short-term memory (LSTM) layer.

The item-item collaborative filtering approach captures the items of interest in a given session by computing item similarities from a session-item matrix containing the implicit feedback. For the user-user collaborative filtering approach, a more sophisticated method is used to compute session similarities. The content-based approach recommends the most similar items to the ones seen in the current session, by computing a distance between two items. Finally, the LSTM approach is based on an idea borrowed from a similar method in the NLP domain for predicting the next word of a sentence. In our case, we will predict which items will be most probably purchased next based on the seen items of a given session. After representing the sessions as a series of considered products, two types of embeddings will be formed: one using the Word2Vec algorithm, and another one using the item's contents.

All approaches at the end recommend the top-ranked 100 candidate items to be bought. It was concluded that the approach with the highest accuracy in predicting the top 100 items was the item-item collaborative filtering.

1 INTRODUCTION

The Reccommender System Challenge (also known as RecSys Challenge) first occurred in 2010 and is a yearly recurring competition, where participating teams can try to find the best-performing recommendation approach for a given scenario. The organizers of the challenge present both a data set and a problem, which the teams later try to solve. The 2022 year Reccommender Systems Challenge data is provided by Dressipi – the fashion-AI experts. Dressipi provides both product and outfit recommendations to the leading global retailers. Recommendations from Dressipi work with brands from 3 continents and allow retailers to create new product discovery experiences as well as make it possible that they are visible throughout the customer's whole shopping journey. This is why this year's task is focused on fashion recommendations – namely to predict which item will be most likely bought at the end of the session by the user, knowing the following variables: user sessions, purchase data, and content data about items. This predicted item should be presented to the user and ideally result in a purchase. The task is to find 100 predictions for every query session in the test set.

2 CHALLENGE TASK

The challenge focuses on fashion recommendation. Participants in the challenge should try to use algorithms, in order to correctly predict the fashion item that will be bought at the end of the session, given the user sessions, purchase data and content data about items. This will enable retailers to predict product demand and size ratio more accurately, and make better purchase and marketing decisions.

The descriptive labels for the items (i.e color, length, neckline, sleeve style, etc.)make up the content data. It is anticipated to be a dataset of high accuracy and quality because the labels were given utilizing Dressipi's human-in-the-loop approach, where fashion professionals examine, revise, and certify the validity of the labels. [1]

To provide the greatest experience that leads to a purchase, it's essential to be able to provide suggestions that are responsive to what the user is doing during the current session. What we need to do is to optimize and deal with the nuances of fashion.

This competition is a typical top-k recommendation competition, but different from ordinary recommendation competitions, this competition does not directly give the user's id, but directly gives a list of the products that the user browsed in one day. We should recommend 100 products that users are most likely to buy based on the list of products that users browsed that day. This requires us to model the user browsing sequence well, which is very beneficial to our entry recommendation competition.

3 DATA

3.1 Data description

The 2022 year Recommender Systems Challenge data, provided by the aforementioned Dressipi, contains descriptive labels of fashion items. This includes length, color, sleeve style, and so on. Data providers claim that the dataset is of high arecommendationsality. The dataset itself consists of 1.1 million online retail sessions collected over an 18-month period, that resulted in a purchase. Moreover, every single position is labeled with content data.

The data is already split into two sets by the date - the training set, consisting of sessions of users who already made a purchase, and the test set. The first one consists of the data for the first 17 out of 18 months, while the latter one is the last month of this period.

The data is given in three files:

- `train_sessions.csv`
  This file contains all the items seen in one session, but which were not bought. Furthermore, a timestamp describes when the item was seen.
- `train_purchases.csv`
  This file contains the items purchased in each session. Furthermore, a timestamp describes when the item was bought.
- `item_features.csv`
  This file contains the item features in a normalized way. This means that for each item and feature id, one row indicates a given value ID. This could for example describe the color of a pair of trousers or some indication of how a skirt is cut.

Participants are additionally informed about the following:

- Approximately 51% of the visitors are new. This is why the solution should be based on the current session activity only.
- Historical data might not be accurate anyway, since in this domain the trends change more rapidly than in others.
- Predictions shall be done as soon as possible in order to handle a very short session and to avoid users bouncing.

3.2 Data preprocessing

Data preprocessing is a technique used in data mining to transform raw data into a useful and efficient format; in our case, we preprocessed our data, so we could use it to make the predictions required in the challenge task. For all our approaches the initial, general data processing was the same. These processing steps are described here.

	item_id	feature_category_id	feature_value_id
0	2	56	365
1	2	62	801
2	2	68	351
3	2	33	802
4	2	72	75
...	...	...	...
471746	28143	68	351
471747	28143	55	390
471748	28143	11	109
471749	28143	73	91
471750	28143	47	549

471751 rows × 3 columns

Fig. 1. Item_features dataset composition and dimensions

We start with the given datasets of "item_features.csv", "train_purchases.csv" and "train_sessions.csv". In 1, you may find a snippet from "item_features" dataset.

There are 471751 rows in the items dataset, however, only 23691 items - in the dataset, one item appears in more than one row when it has more than one feature. This dataset was denormalized so that we have all the information for each item in one row. This leads to many null values in the denormalized table, as a single item has only very few properties (e.g. trousers have different properties than skirts or t-shirts).

We followed the same procedure with the session dataset to see its dimensions (4743820 rows × 3 columns).

Next, we combined the views inside a session and the purchases of this session in one dataframe, with the column "was_bought"; this way we could indicate which items were viewed or bought in a given session. Then we merged the denormalized items table with the combined dataset with session views and purchases (that was previously modified), so we could have a more useful representation of the item features. After replacing the null values with 0 we get a representation of the data as in 2.

For some of the approaches, additional specific data preparation was needed. These steps are described in the next section, where each approach is treated in a more detailed way.

4 APPROACH

In this project, the following main approaches were chosen:

4.1 Item-item collaborative filtering

The first step was to prepare the data for item-item collaborative filtering similar to assignment 1 from the course. For this we did also include the test data, in order to be able to do the predictions in the end for these test sessions. In order to handle implicit feedback, we have set the rating of bought items to 1 and the rating of seen but not bought items to

	session_id	item_id	date	was_bought	item_feature_0	item_feature_1	item_feature_2	item_feature_3	item_feature_4
0	3	9655	2020-12-18 21:19:48.093	0.0	NaN	53.0	NaN	NaN	NaN
1	3	9655	2020-12-18 21:25:00.373	0.0	NaN	53.0	NaN	NaN	NaN
2	3	15085	2020-12-18 21:26:47.986	1.0	NaN	53.0	NaN	NaN	NaN
3	13	15654	2020-03-13 19:35:27.136	0.0	NaN	NaN	NaN	618.0	NaN
4	13	18626	2020-03-13 19:36:15.507	1.0	NaN	NaN	793.0	618.0	605.0
...	...	...	...	...	...	...	...	...	...
5743815	4440001	19539	2020-10-30 23:37:09.46	0.0	NaN	NaN	NaN	618.0	NaN
5743816	4440001	20409	2020-10-30 23:37:20.658	0.0	NaN	NaN	NaN	618.0	NaN
5743817	4440001	27852	2020-10-30 23:39:55.186	0.0	NaN	NaN	NaN	618.0	NaN
5743818	4440001	20449	2020-10-30 23:40:28.149	0.0	NaN	NaN	NaN	618.0	NaN
5743819	4440001	16631	2020-10-30 23:46:05.218	1.0	NaN	53.0	NaN	NaN	NaN

Fig. 2. Dataset after general preprocessing

0.5. The idea behind this is that seen items shall have a higher rating then non-seen items (which were not at all of interest in the current session). This way, we capture how interesting an item was with respect to a given session.

In this manner, we obtain a session-item matrix containing the implicit feedback. As indicated in lecture 4 about implicit feedback (slide 8), it makes sense to normalize user vectors when working with implicit data. This was implemented as well. From the resulting session-item matrix containing the implicit feedback, we computed a rating prediction (i.e. prediction of implicit feedback). This is done according to lecture 10 about sequence-aware recommenders (slide 14): The "prediction for a target item is the sum of similarities of all current session items".

To achieve this, we wrote a function computing the item-similarities in the fashion of assignment 1. To make our computations faster, we pre-compute the item-similarities for the candidate items and store them in a dictionary.

Finally, for a given session and each candidate item the predicted rating (i.e. predicted implicit feedback score) is computed as described above. This score will then be used for ranking our candidate items, from which the top 100 are extracted.

4.2 Content based

In this approach, we tried to recommend the most similar items to the ones seen in the current session. For this, we define a distance function between two items: the computed distance is simply the number of non-equal features (as the feature values are only categorical). For each candidate item and each item in the current session, the value of this distance is pre-computed so that the computation in the recommendation step is faster. In the recommendation step, for each candidate item and each item in the current session, this distance is retrieved from the from the dictionary of pre-computed distances. In the end, all those distances will be summed up for the current session. Thus, if a session has k items, we retrieve for each candidate item k distances and then take the sum of these k distances. This way, we get a distance between the whole current session and each candidate item.

Finally we order the candidate items by ascending distance between the current session and candidate item, and retrieve the top 100 candidate items. This way, we will get the closest items to that session.

4.3 LSTM based

Similarly to predicting the next word of a sentence in natural language processing, we can predict the "next item" of a session. For this, we see the viewed (but not purchased) items as a given sequence of input words, and the purchased items as output words (i.e. next words). This is often done by a long short-term memory (LSTM) neural network model.

In order to be able to use the LSTM model, we first need to prepare the training data for the LSTM. We selected only a subset of the training dataset, so that RAM is not exceeded. We extracted the viewed items (corresponding to the previous words) and the purchased items (corresponding to the next words). The viewed items form a sequence of "words" which will give rise to the next occuring "word" (item). The relationship between this sequence of input items and the purchased next item shall be caputured by our LSTM model.

Thus, the input of our model is always a sequence of items observed in a session. However sessions do usually have different lengths. Therefore we take the longest session as a standard session length and pad the shorter sessions with zeros in the beginning.

Our LSTM model needs an embedding, so that each item can be represented by a vector of numeric values. For this, we tried two different strategies to come up with an embedding.

(1) **word2vec**: To represent each item by some feature vector, we use Word2Vec embedding.
(2) **contents**: To represent each item by some feature vector, we use the item contents as an embedding. This means that we encoded the contents of the items (one hot encoding) and used this encoding for our embedding.

In the end, our LSTM model contains

- an Embedding layer according to the strategies described above,
- an LSTM layer with 256 units
- a Dropout layer with a ratio of 0.2 (in order to avoid overfitting)
- a Dense layer with 128 units and relu activation
- a Dense layer with as many units as occuring words (items) and a softmax activation.

The model was trained with 10 epochs.

In order to predict with our LSTM model, it was needed to prepare the data in the same fashion as described for the training set above. In order to not exceed the RAM, we have split the test dataframe into sub-dataframes (batches of sessions), which will be predicted batch by batch. After prediction, we obtain for each candidate item and each given session a score for this candidate item. From this, we extract the 100 items with the largest scores.

4.4 User-user collaborative filtering

The User-user collaborative filtering is done with the assumption that the sessions could be seen as (different) users - the idea is from Sequence-Aware Recommenders, slide 13 - therefore they can be compared. The comparison is based on the features of the items in the sessions.

The first step to make them technically comparable is the introduction of a new "categorized feature" attribute. Which is basically an addition of the feature category ID (multiplied by 10000) and the actual feature value. Then the "train" and "target" data sets should be created with the session_id and categorized_feature columns. The "target" data

set should contain the sessions to be predicted. This two data set should be outer-joined on the "categorized_feature", and then grouped on the target session_id.

So the similarity between two users can be calculated as "CCFta" / ("CFtr" + "CFta" - "CCFta") where

- CCFta is the count of the common distinct categorized features between "target" and "train" sessions
- CFtr is the count of the distinct categorized features in the "train" sessions
- CFta is the count of the distinct categorized features in the "target" session

After the similarity calculated between the "target" and "train" sessions, the top 100 similar train sessions can, should be selected. In these sessions it is also stored which item was purchased therefore these items are the candidate items, ranked by the similarity.

5 CHALLENGES

The most important challenges during the project were:

- Dealing with implicit data
- Huge data size leads to long computation times
- Huge data size leads to memory issues
- Data sparsity: Only about 5 features per item but 904 values per vector. Sparse data format, embedding layers.
- Session-based recommendation where user profiles are invisible and there is no historical data for about 51% of all users
- Important to give recommendations that respond to what the user is doing during the current session.
- Sessions can be pretty short.
- Even for users with historical data available, fast fashion trends and other extraneous factors out-dates user preferences much quicker than in other domains.
- No rating matrix. Instead, only sequentially ordered log of user interactions is provided: purchased/observed/non-observed.

5.1 Major Challenges for RS in the fashion domain

The variety of information sources used by Recommender Systems (RS) can extend beyond the user-item matrix, usually given by qualities or information related to item category, such as color, texture, style. It is crucial to learn from existing user interactions, comprehend, and discover the underlying decision factors in order to produce useful recommendations. Understanding fashion apparels enables a solid foundation upon which recommendations can be built.

In this section, the major challenges faced by RS in the fashion domain are described.

Fashion-item representation. Due to the sparse purchase data/information about the visual appearance of the product in category names, traditional Recommender Systems, like Collaborative Filtering or Content-Based Filtering, struggle in the fashion sector [8]. Additionally, recent literature has made use of models that capture a rich representation of fashion items through product images [5], text description [2], which are frequently learned through classification, product retrieval etc. Nevertheless, learning product representations from this types of input data, would require large sets of data to generalise effectively over various image/text styles, attribute/feature variations etc. Moreover, it is still an open-research question how to build a representation that learns which product attributes consumers value most when assessing fashion items.

Fashion-item compatibility. It's a challenging task to train a model that can estimate whether two fashion items go together or create one outfit by combining several items.

Fig. 3. Predicting items that go together

Some signals that can be used for item compatibility are co-purchase data [7], outfits designed from professional fashion designers [3], or combinations from social media photos [6]. This compatibility evaluation together with the associated pictures and text data is used to learn to predict similar products. For instance, Zhou et al's CNN implementation [4] is one solution to the problem of matching two-piece clothing to current fashion trends. To allow the system to distinguish garment characteristics, they combined perceptual and reasoning models and built two parallel CNNs, one for clothes worn on the upper body and the other for clothing worn on the lower body. The generated data is incorporated into style themes using a hierarchical topic model, which has superior semantic interpretation of the collocation patterns. Furthermore, another challenge in the prediction of compatible items is the dependency on trends, seasonality, brands or location.

Discovering trends. By forecasting consumer preferences, retailers may improve product-to-market fit, advertising and logistics. Which characteristics are deemed "fashionable" or "trendy" depends on a variety of variables, such as seasonality, location or style dynamics. Once more, researchers make good use of social media.

The aforementioned challenges, and many other, are what we came across throughout our research for the project, also mentioned in studied research works.

6 RESULTS

To compare our models, the models were executed on the leaderboard dataset provided by the Recommender Systems challenge. The results were uploaded there to get a metric.

Unfortunately we could not include the results from our user-user collaborative filtering method because of the occurred memory and computation time issues we could not solve. Therefore this UU-CF is working for one or a couple "target" sessions, but definitely not for 50k.

In the end, the best models were the item-item collaborative filtering approach with an MRR score of 0.090 and the content based approach with an MRR score of 0.047. The best scores of the challenge being 0.216 with probably very sophisticated approaches, we think that our final result is decent for a first recommender systems project.

For the LSTM models we have seen that the item coverage is rather low and the same items are recommended over and over again.

7 CONCLUSION

All in all, we can conclude that the methods introduced in the lecture provide suitable recommendations can be done, even for complex underlying topics, such as fashion recommendation. But dealing with such a big dataset can be problematic with limitations on the environment.

8 DISTRIBUTION OF WORK

Our team was rather a loose formation, we did some brainstorming together, but basically we worked mostly alone/in pairs. Nevertheless the preparation of the datasets, and most of the approaches were done by Julian Backé.

REFERENCES

[1] Bruce Ferwerda, Saikishore Kalloori, Abhishek Srivastava. Recsys challenge 2022. http://www.recsyschallenge.com/2022/#about. Accessed: 2022-07-09.

[2] Xu Chen, Hanxiong Chen, Hongteng Xu, Yongfeng Zhang, Yixin Cao, Zheng Qin, and Hongyuan Zha. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 765–774, 2019.

[3] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S Davis. Learning fashion compatibility with bidirectional lstms. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1078–1086, 2017.

[4] Xing Hao, Han Zhike, and Shen Yichen. Clothnet: A neural network based recommender system. *Fuzzy Systems and Data Mining VI: Proceedings of FSDM 2020*, 331:261, 2020.

[5] Ruining He and Julian McAuley. Vbpr: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[6] Shatha Jaradat. Deep cross-domain fashion recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 407–410, 2017.

[7] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

[8] Dandan Sha, Daling Wang, Xiangmin Zhou, Shi Feng, Yifei Zhang, and Ge Yu. An approach for clothing recommendation based on multiple image attributes. In *International conference on web-age information management*, pages 272–285. Springer, 2016.