Hotel Recommendation System Using Deep Matrix Factorization for Implicit Feedback Data

Joshua Rosairo
The University of Auckland
Auckland, New Zealand
jtha772@aucklanduni.ac.nz

Manuel Agüero
The University of Auckland
Auckland, New Zealand
jagu688@aucklanduni.ac.nz

Maria Soleim
The University of Auckland
Auckland, New Zealand
msol457@aucklanduni.ac.nz

Rifky Rismon
The University of Auckland
Auckland, New Zealand
rmuh218@aucklanduni.ac.nz

***Abstract*—Recommendation systems are one of the most successful and widespread applications of machine learning technologies in the e-commerce industry. Here, Alternating Least Square (ALS) based Collaborative Filtering (CF) methods implementing a Matrix Factorization (MF) approach are one of the most popular recommendation systems models. However, this technique has it has a major drawback: the ALS algorithm is unable to handle the cold-start problem, that is, properly integrating users newly added to the system; thus, the user past behaviour appears as missing entries in the dataset. We address this problem by implementing a collaborative filtering algorithm with a deep learning approach, called deep matrix factorization, this approach allows us to integrate the users implicit feedback and side information data to optimize the accuracy of the model. To make our analysis, the dataset provided by Trivago as part of the RecSys 2019 challenge was utilized.**

***Index Terms*—alternating least square, collaborative filtering, recommendation systems, deep matrix factorization, implicit feedback**

I. Introduction

In today's dynamic business environment, one of the biggest challenges that the e-commerce industry faces is finding ways to provide a personalized recommendation that enhances customer experience and engagement. Recommendation systems provide a solution to this problem by predicting user preferences for certain items through the use of a rating system. This allows for the creation of lists of recommendations tailored to the user, ensuring that only those elements attractive for the user are provided.

Collaborative filtering (CF) is one of the most extensively developed models for recommendation systems on e-commerce. The collaborative filtering approach is based on the assumption that, if users displayed similar interests in the past, they are likely to agree in the future [1].One of the most widely-used algorithms for implementing collaborative filtering is *Matrix Factorization* (MF); Here, explicit feedback data from user-item interactions, derived from historical behaviour, is utilized to build a prediction engine [2]. Values representing each user interests are assigned in relation to certain items. In particular, when explicit feedback is provided, latent factors are extracted from those values to estimate the likelihood of an user to prefer a specific item, thus, modeling users potential interests. One of the techniques that can be implemented to extract such that latent factors is called *Alternating Least Squares* (ALS).

However, every technique has its corresponding drawbacks, in the case of the ALS method the problem of handling the introduction of new users or items to the system with no previous information about them arises. This problem is referred to as the *cold start problem* [3]. As a consequence, when information about users and items is limited , the ALS method is unable to properly handle the sparse dataset, leading to the limitations in the ability to produce proper item recommendations to the users. On the other hand, every user generates vasts amounts of information by every interaction at every step of their experience. Precious and valuable information can be extracted by monitoring activities like purchasing behaviour or analyzing a wide variety of data such as filter selection and number of clicks, frequently viewed items or even the user location. We refer to this type of information as *implicit feedback* data due to the complicated nature of extracting meaningful content from this data [1]. The use of of this kind of vast information about the user behaviour and preferences has the potential to provide a significant improvement to the results obtained by implementing a traditional matrix factorization alternative.

Therefore, In this paper we have implemented a novel algorithm denoted *Deep Matrix Factorization* (DMF). Through the use of deep learning, this approach facilitates the integration of side information for effectively deriving more accurate predictions when interacting with the implicit feedback data provided as input [4]. The input data is utilized to transform the sparse high-dimensional data low-dimensional data containing a flexible variety of side information, this processed is called *implicit feedback embedding*. This newly embedded information is then used for the the extraction of the latent factors, the different layers of this framework are in charge of executing this process. Hence, in this paper we compare the performance of the Deep Matrix Factorization model with Implicit Feedback Embedding with that of the ALS method in order to achieve the best recommendation and increase model training efficiency compared to conventional traditional methodologies.

This paper is organized as follows: In the next section, we

will outline our hypothesis. Then, the previous works regarding the DMF algorithm will be discussed. In the following sections, the methodologies for testing our hypothesis with the methods previously described will be presented. Finally, our results and future work will be covered in the remaining sections.

II. Hypothesis Description

In this paper, our primary objective is to build a recommendation system that is able to predict which accommodations have been clicked during a certain user's session. Furthermore, we propose a novel algorithm called Deep Matrix Factorization with Implicit Feedback Embedding to deal with a cold-start problem (often encountered in real-world scenarios) by integrating the implicit feedback data and the side information of the user's behaviour as an input data and using the implicit feedback embedding for the pre-processing technique. On top of that, the deep learning model will be used to extract the latent factors from the user-item matrix embedding to predict the item with the largest click count.

We hypothesize that the combination of using the pre-processing technique, leveraging the user's side information and deep learning model, will allow the model to gain a higher accuracy than the one achieved by our baseline model: Alternating Least Squares.

III. Previous Works

A. The Importance of Implicit Feedback

Since implicit feedback does not provide user preferences directly, it is of great importance to select the variable that most accurately represents the the users interests with respect to a set of items. Yifan Hu, et al [1] described several prime characteristics to be searched for in the implicit feedback data and emphasize the importance of selecting this kind of information for building a recommendation system.

A big factor for highlighting the importance of implicit feedback data is when there is no negative feedback in the dataset. For instance, a user who didn't consider a hotel might not have done so due to a lack of information or the inability to access that item for consideration. Specific recommendations tend to concentrate on the data collected from user-item pairs we know about; their interactions are explicitly provided. However, in most cases the remaining user-item interactions has yet to be described, this typically corresponds to the the vast majority of the data, and is tends to be treated as "missing entries" to be omitted from the analysis. When most of the user-item information is not explicit, treating unrepresented interactions as missing entries would only leave us with the small amount of explicitly collected feedback, leading to a sever misrepresentation of the complete user profile. It is therefore essential for the missing information to be taken into consideration as well, This is also where most negative feedback is expected to be found.

Another reason for using implicit feedback information is the following: while the explicit feedback numerical value shows preference, the implicit feedback data the numerical value shows a confidence level. Explicit feedback-based systems allow users to express their preference level, such as a star rating between 1 ("totally dislike") and 5 ("really like"). On the other hand, implicit feedback numerical values describe the frequency of actions, e.g. how long the user watched a certain show, how often a user purchases a certain item, etc. There is no greater preference for a bigger value. For example, the hotel that a user liked the most might be one that is not frequently considered due to, say financial limitations; meanwhile a hotel with low that displays low preference by the user might be be the one that the user has the most interaction with, perhaps for convenience. Thus, implicit feedback information encompasses extremely valuable data essential for accurate user behaviour predictions.

B. ALS Collaborative Filtering for Implicit Database

While the selection of implicit data is vital for the formulation of a recommendation system, another important factor to consider, is the using the right algorithm to construct for the model implementation. The Alternating Least Squares Collaborative Filtering based algorithm for implicit feedback data was developed by several researchers and has been previous work utilizing this model has been done [1] [5]. ALS is an iterative optimization process where a factorized representation of the original is approximated to a greater degree at each iteration [5].

The evaluation methodology used in these papers [1] [5] [5]. utilizes an ordered list of items sorted from the main prediction defined as the most preferred to the least preferred one; a similar evaluation process is necessary for our problem formulation. The results yielded from a naive recommendation measure and a neighbourhood based measure were compared to the ones produced by this model; such measures were found to be significantly less accurate.This model allows for different methods to be implemented in order to derive the user preferences from the confidence levels obtained by applying different threshold values. Furthermore, this model aims to balance the extraction of properties from implicit feedback datasets and computational scalability, thus leaving room for improving the accuracy at the expense of an increase in computational complexity.

One important point to highlight from these papers is that choosing the number of latent factors and regularization parameters are highly relevant in optimizing the accuracy of the model. For instance, in [1], an accuracy lower than lower than their baseline model (popularity based model) was achieved when using 100 factors and a parameter $\lambda = 0$.

However, the accuracy of the ALS algorithm can outperform the baseline model using using 50 factors and $\lambda = 500$. On top of that, they highlighted the inability of this algorithm to handle a cold start problem for users newly to the system.

C. DMF with Implicit Feedback Embedding (IFE)

Baolin Yi, et al. [4] proposed a deep learning based Collaborative Filtering model called deep matrix factorization. This model intends to facilitate the integration of side information in an effective manner for deriving accurate predictions when interacting with the implicit feedback provided as input. One of the most popular approaches for CF is based on low-dimensional matrix factorization models which work by building a feature extraction model for side information available and training the matrix factorization utilizing such features as the prior of the latent factors. Several deep learning methods have been employed to estimate this prior.

This paper aims to enhance those methods by proposing an implicit feedback embedding that transforms high-dimensional sparse feedback information to the low-dimensional real value vectors required for the matrix factorization methodologies. Then DMF structures two featured transforming functions implemented with a multi-layer perceptron to generate the latent factors used for prediction. DMF reduces the scale of model parameters and increases model training efficiency obtaining better performance than traditional methods. However, the preparation and tuning process of a deep learning model is complicated. Furthermore, DMF model utilizes a very simple encoding for the side information and takes a heuristic approach for parameter determination. Hence, the optimization of this model is an area open for discussion. Evaluation metrics that this paper used are the mean absolute error (MAE) and the root mean square error (RMSE). The smaller values of the MAE or RMSE means that the recommendation system has a higher the accuracy.

In their experiments, they used five different datasets: MovieLens-100k, MovieLens-1M, Douban-Book, Douban-Movie, and Douban-Music dataset. They compared the performance of the DMF model against other five algorithms. These five algorithms are *Mean Method, Probabilistic Matrix Factorization, AutoRec, Neural Autogregressive Distribution, Deep Learning for Long Tail Web Service Recommendation,* and *Representation Learning via Dual-AutoEncoder*. The DMF model consistently outperformed all the other algorithms for all the five different datasets. For example, in the MovieLens-1M dataset, the RMSE and MAE for the DMF model is 0.8321 and 0.6082 respectively. However, *Neural Autogregressive* Distribution model come close with a RMSE and MAE value equal to 0.8984 and 0.6578. This experiments proved that, using a deep learning approach to leverage the implicit data and side information, we can enhance the recommendation system performance.

IV. Research Methodology

A. Datasets

The dataset used is provided from RecSys 2019, an annual recommendation system challenge. This year (2019), the challenge is hosted by Trivago, a technology company specializing in offering an extensive hotel meta search for travellers [6]. The challenge is to develop a recommender system able to predict which accommodations that have been clicked in the search result during the last part of an user session.

Four different datasets are given: train, test, item_metadata, and submission popular and they are all given in csv format. The training dataset has around 15 million records, which contains end-to-end users activities within a session before the user clicked one of the items from the recommendation that appeared on the website. The behaviours in the data are uniquely identified by their user_id, session_id, timestamp, and step. The variable action_type defined in details what interaction a user did on the website, such as interaction item rating, interaction item info, interaction item image, change of sort order, and click out item. Figure 2 is a snapshot of our training data.

user_id	session_id	timestamp	step	action_type	reference	platform	city	device	current_filters	impressions	prices
00RL8Z8Z2Z21	aff3928535f48	1541037460	1	search for poi	Newtown	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037522	2	interaction item image	666856	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037522	3	interaction item image	666856	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037532	4	interaction item image	666856	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037532	5	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037532	6	interaction item image	666856	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037532	7	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037532	8	interaction item image	666856	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037542	9	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037542	10	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037542	11	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037542	12	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037542	13	interaction item image	109038	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541037543	14	clickout item	109038	AU	Sydney, Australia	mobile		3400638|12 95|66|501|1	
00RL8Z8Z2Z21	aff3928535f48	1541038469	15	search for poi	Surry Hills	AU	Sydney, Australia	mobile			
00RL8Z8Z2Z21	aff3928535f48	1541038485	16	clickout item	1257342	AU	Sydney, Australia	mobile		55109|1293 162|25|150	
02HGRBA06ODU	fa3a53156ae5c	1541030832	1	interaction item info	3377332	BR	Ubatuba, Brazil	mobile			
02HGRBA06ODU	fa3a53156ae5c	1541030843	2	interaction item image	3377332	BR	Ubatuba, Brazil	mobile			
02HGRBA06ODU	fa3a53156ae5c	1541030843	3	interaction item image	3377332	BR	Ubatuba, Brazil	mobile			

Fig. 1. Snapshot of the training data.

The testing data set, which contains 3 million records is similar like the training dataset, except the click out items is blank as we are going to predict that clicked out time using our recommendation model. Below we show a snapshot of the testing data.

user_id	session_id	timestamp	step	action_type	reference	platform	city	device	current_filte	impressions
004A07DM0I	1d688ec1689	1541555614	1	interaction item image	2059240	CO	Santa Marta	mobile		
004A07DM0I	1d688ec1689	1541555614	2	interaction item image	2059240	CO	Santa Marta	mobile		
004A07DM0I	1d688ec1689	1541555696	3	clickout item	1050068	CO	Santa Marta	mobile		2059240|2033381|1724779|127131|399441|103357|127132|
004A07DM0I	1d688ec1689	1541555707	4	clickout item	1050068	CO	Santa Marta	mobile		2059240|2033381|1724779|127131|399441|103357|127132|
004A07DM0I	1d688ec1689	1541555717	5	clickout item	1050068	CO	Santa Marta	mobile		2059240|2033381|1724779|127131|399441|103357|127132|
004A07DM0I	1d688ec1689	1541555792	6	clickout item	3241426	CO	Santa Marta	mobile		2059240|2033381|1724779|127131|399441|103357|127132|
004A07DM0I	1d688ec1689	1541555799	7	clickout item		CO	Santa Marta	mobile		2059240|2033381|1724779|127131|399441|103357|127132|

Fig. 2. Snapshot of the testing data.

Based on the figure above, there is a missing clicked item for user 004A07DM0I which we are going to predict. The prediction of the clicked out item must be between the items that appear in impressions list. Therefore, we will predict the score of each users for all their impression list and sort the obtained item scores from the largest to smallest. The largest

score represents the item with the highest likelihood of being clicked by the user.

The training data will be used to build the recommendation systems and the accuracy will be tested using the testing data. The dataset item_metadata contains all the item_id available in the data and have a list of filters that are applicable for the given item. The last dataset, submission popular gives the delivery format for the participants in the RecSys challenge.

One crucial problem that appears in the dataset is that more than 90% the user_id in the testing dataset does not appear on the training dataset. It means that this user_id are new to the Trivago website. Therefore, in this dataset, we have a cold start problem mentioned in the introduction chapter.

B. ALS Collaborative Filtering Recommendation System

ALS was chosen to be the baseline algorithm for the project. Usually, this method is used on explicit feedback. However, in this research the dataset is implicit. Instead of using a matrix giving explicit scores between user and movies, an user-item matrix is created where the value 1 is given for a user-item pair if the user has performed a click out on that item and value 0 otherwise.

Two new lower dimensional matrices are initialized, denoted the user matrix and the item matrix. In the user matrix the number of rows equals to the number of users in the original user-item matrix, while the columns are so-called latent factors which represent the items in a much lower dimensional space than the original matrix. Similarly, the number of columns in the item matrix is equal to the number of items in the original matrix, and the rows are latent factors. The the dimension of the latent factors is equal in the user and item matrix so that they can be multiplied together to form the original matrix, as shown in figure 3. In the figure, the latent factor is 2, but in this research, the dimension of the latent factors was set to 20.

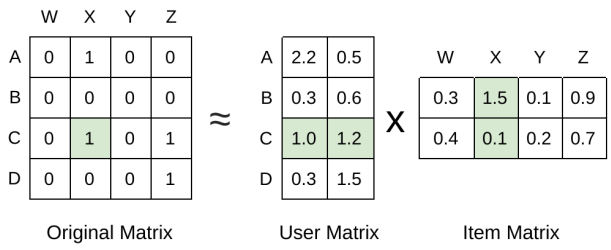


Fig. 3. A high-dimensional user-item matrix is factorized into two lower-dimensional matrices.

The matrices are initialized randomly. Then the algorithm alternates holding one matrix fixed while running gradient descent with the other matrix. The gradient descent works in the following way. A random user-item pair is chosen, say user C and item X. The dot product between the user vector from the user matrix and the item vector from the item vector is computed. This is the predicted score, $\tilde{r}$, which should ideally be equal to the user-item value. In this case we obtain:

$$\tilde{r} = 1.0 * 1.5 + 1.2 * 0.1 = 1.62$$

The error will be

$$e = \tilde{r} - r = 1.62 - 1 = 0.62,$$

where r is the real value. Then, the user vector, $\vec{v}_u$, and item vector, $\vec{v}_v$ is updated one by one by the formulas:

$$\vec{v}_u = \vec{v}_u - \epsilon(error * \vec{v}_i^T + \lambda * \vec{v}_u)$$
$$\vec{v}_i = \vec{v}_i - \epsilon(error * \vec{v}_u^T + \lambda * \vec{v}_i)$$

For example, if ϵ = λ = 0.01, we have

$$\begin{aligned}\vec{v}_u &= [1.0, 1.2] - 0.01 * (0.62 * [1.5, 0.1] + 0.01 * [1.0, 1.2]) \\ &= [0.9906, 1.1993]\end{aligned}$$

$$\begin{aligned}\vec{v}_i &= [1.5, 0.1] - 0.01 * (0.62 * [1.0, 1.2] + 0.01 * [1.5, 0.1]) \\ &= [1.4937, 0.09256]\end{aligned}$$

If we now calculates the new predicted value for user C on item X, we will get 1.59 which is closer to 1 than 1.62. This procedure is repeated until convergence.

C. DMF Recommendation System

Matrix factorization models tend to utilize explicit rating information to build an user-item matrix that represents their interaction. However, in the Trivago dataset there is no item rating associated to an specific user. Hence, since we utilize the user-item click-out data as an alternative to the ranking entries in the user-item matrices.

An undirected graph G represented by a matrix was constructed, where the set of users and set of items are connected by edges that represent the historical behaviour between them; that is, the implicit feedback data. Next, the user and item space, along with the edges representing their relationships, is to be projected into a vector in a $k - dimensional$ space. In contrast to the traditional matrix representation, this approach reduces the parametric complexity scale involved in the subsequent stage. Negative sampling is introduced to generate additional feedback.

We denote c_{ij} as the number of click-out interactions that the user i has with respect to the item j. Hence, our model aims to rank all items based on likelihood of a click-out interaction to occur; the number of predicted click-out interactions generated is then represented as $\hat{c}_{ij}$.

Now, the user-item information received as input consists of sparse high dimensional data. Thus, an implicit feedback embedding was implemented to transform the high-dimensional data into low-dimensional real value vectors.

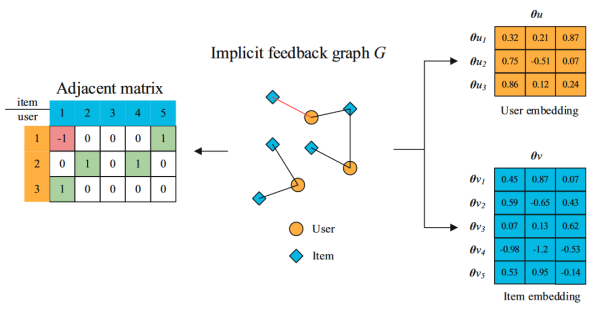


Fig. 4. From "Deep Matrix Factorization with Implicit Feedback Embedding for Recommendation Systems" [4]

The TensorFlow platform [7] was utilized for performing the corresponding embedding. An user i and item j bias is incorporated as an unknown vector of length K yielding the corresponding embeddings $\theta^e_{uij}, \theta^e_{vij}$. This vectors are assigned random initial values

The user and item information embedding output dimensions have no inherent meaning, however the deep matrix factorization model uses the patterns encountered in the generated low-dimensional data to derive the latent features required for prediction. Furthermore, due to the large amount of data, traditional deep learning models require an unfeasible computational time when resources are scarce. By embedding the user-item data the computational complexity required for training the model is greatly reduced.

An schematic view of the implicit feedback embedding is presented below:

Once the embedding of the user and item information have been completed the side information is integrating using a one hot encoding and concatenation operation to the respective user-item vectors, the click-out prediction can be computed by calculating the dot product of the two embeddings:

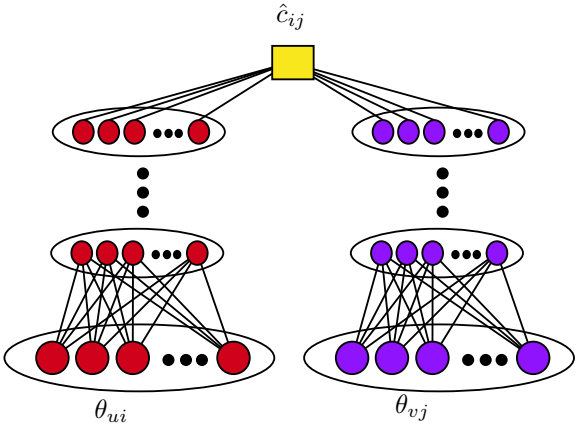
$$\hat{c}_{ij} = \theta^b_{ui} + \theta^b_{vj} + \sum_{k=1}^{K} \theta^e_{uik}\theta^e_{vjk}$$

where $\theta^b_{ui}, \theta^b_{vij}$ represent corresponding user and item bias.

The goal is to minimize the associated loss function, with hyper-parameters μ_u and μ_v to incorporate the generalization capabilities characteristic of DMF models:

$$L = \sum_{(i,j)\in G} (c_{ij} - \theta^b_{ui} - \theta^b_{vj} - \sum_{k=1}^{K} \theta^e_{uik}\theta^e_{vjk})^2$$
$$+\mu_{\theta_u} \sum_{i,k} (\theta^e_{uik})^2 + \mu_{\theta_v} \sum_{j,k} (\theta^e_{vjk})^2$$

The feature transformation process is carried out by a multi-layered perceptron with the goal of deriving the probability distribution from the input features. In ordered to make this implementation efficient and memory scalable a mini batch gradient descent training step is implemented. A batch size of 200, learning rate of 0.02 and training duration of 50 epochs were heuristically chosen. The capacity of the model is controlled by the structure and the number of hidden neurons in the multi-layered perceptron. A schematic view is shown below:



D. Model Evaluation

The RecSys challenge provided a hidden dataset with the click outs that were missing in the test set, to test the performance of their participants algorithms. To evaluate the results in this research, this advantage was exploited. A submission file with up to 25 predicted click out items per missing value were created. The RecSys challenge used the mean reciprocal rank (MRR) to evaluate the submission. The mean reciprocal rank is a statistical measure for assessing any method that generates a list of possible answers to a sample of queries, ordered by correctness probability [8]. The formula for calculating the MRR is described as follows:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

where rank-i refers to the rank position of the actual clicked accommodation for the i^{th} click out item. For each missing click out item, the number one is divided by the correct items position in the predicted list. An item earlier in the list will give a higher value than an item later in the list. Then the mean score over all the predictions is taken, and that is the MRR. The following examples are given by the RecSys 2019 website [6].

query 1:
impressions = [100, 101, 102, 103, 104, 105]
clicked_item_id = 102
submission = [101, 103, 104, 102, 105, 100]
reciprocal rank = 0.25

query 2:
impression = [101, 103, 104, 100, 105]

clicked_item_id = 105
submission = [103, 105, 101, 100, 104]
reciprocal rank = 0.5
mrr = (0.25 + 0.5) / 2 = 0.375

When evaluating this way, it is important to avoid peeking [9]. Peeking occurs when several different approaches are taken, for example running an algorithm with different parameters and check the score for each of them, and the best one is chosen. In this way information leaks into the learning algorithm such that the algorithm over-fits to the test set, and wont generalize well when it is used on a new dataset. For this reason, this approach is used only for the final steps in evaluating the algorithms.

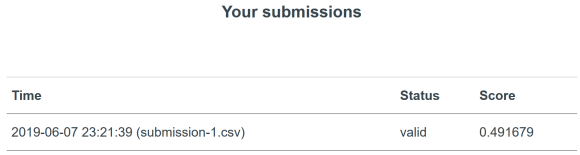


Fig. 5. DMF submission result.

Fig. 6. ALS submission result.

V. Results

A. Results for the Baseline Algorithm

Due to the time constrains and limited computational resources, we were unable to test extensively test this approach to optimize the result yielded by the ALS algorithm. We heuristically determined the following parameters, within our limitations, to obtain the highest possible accuracy: Regularization parameter $\lambda = 1$ and number of latent factor $= 20$. With these parameters the ALS algorithm implemented obtain a score of $MRR = 0.483846$ when submitted to the Recsys platform. A result of $MRR = 0.5$ would indicate that item with ranking in the second place has been selected for the user recommendation. Hence, our result indicates that the solution is always close to the accommodation placed as the second item in the optimal ranking list. We conclude that out baseline algorithm performs well.

B. Results for the DMF Model

The results achieved by the DMF implementation were not significantly different that the ones produced by the ALS approach; a value of $MRR = 0.491679$ was obtained. However, due to hardware limitations the implementation of this model was constrained to a low number of iterations and embedding dimension, which restricts the training capabilities. Thus, with the right equipment, there is potential for improving the results achieved by this method as the accuracy improved substantially when more side information was embedded and the number of iterations was increased when testing smaller samples. The code used for both the DMF and ALS implementations can be found here [10]

VI. Future Work

Firstly, a major challenge for implementing the Deep Matrix Factorization model was the large amount of computational power required. Therefore, great testing opportunities come with an increase of computational capabilities to leverage the DMF approach. Furthermore, the parameter and embedding optimization remains an area of interest that relates to the scalability of deep learning approaches where large datasets are in play. Using different combination of the deep learning parameters would lead to a very different result. Another area of work resides in the processing and selection of relevant side information for the embedding stage. Side information plays an important role to address the cold-start problem. The integration of the implicit feedback and careful processing and selection of side information has the potential to boost the recommendation model's accuracy.

On top of that, due to the nature of the Recsys challenge and its evaluation methodology, our approached was focused on accuracy rather than scalability and explainability; This two factors allow for further development of the analysis of the models implemented in this project. Finally, this paper applied the discussed algorithms uniquely to the hotel recommendation domain; Moreover, the majority of the results cover correspond the context of movie recommendations. Hence, the performance of these models presents an area of opportunity for analysis in different domains where it remains largely unexplored. Finally the results Within this domain can be further expanded by testing with datasets corresponding to different public and private institutions, thus, allowing for a more robust conclusion.

VII. Conclusion

Our baseline model, the ALS algorithm, offers simplicity in computational complexity and lower amount of parameter requirements. In practice, this algorithm is one of the most popular algorithm implemented for recommendation frameworks. However, as previously discussed, the handling of the cold start problem and remains a challenge when using sparse datasets; furthermore, valuable user and item side information is largely ignored. In order to tackle this drawbacks, we proposed the implementation of a deep learning based recommendation framework denoted Deep Matrix Factorization. The use of DMF aims to facilitate the integration of side information in an effective manner for deriving accurate predictions when interacting with the implicit feedback data provided as input. With this

capabilities, the cold-start problem and the data sparsity that is frequently encountered with building and implementing recommendation systems can be addressed. However, DMF demands a large computational power for training the data, thus, further work can be done in making this model more efficient.

In this report, we used the click out count to represent the rating commonly derived by implicit feedback data. We based such decision on the assumption that the user preference for an item directly corresponds to the user interaction with said item, in particular, the number of clicks. Based on our results, we showed that the accuracy of the DMF Model algorithm outperforms the our baseline model ALS algorithm (0.483846 vs 0.491679 respectively) for the dataset provided for the RecSys 2019 challenge. Since this result is obtained even without optimal computational resources, we therefore confirm our initial hypothesis. A significant amount of side information had to be excluded from the DMF embedding; furthermore, a sub-optimal number of iterations was performed, in contrast to the previous implementations of the model in the literature (as discussed in the previous works section). Hence, we conclude that DMF proves to be a promising model for enhancing accuracy when used in recommendation systems; at the same, plenty of opportunity for improving this model exists. Thus, further work in the analysis and implementation of the DMF model is highly relevant to the development of recommendation systems.

REFERENCES

[1] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," in *2008 Eighth IEEE International Conference on Data Mining*. Pisa, Italy: IEEE, Dec. 2008, pp. 263–272. [Online]. Available: http://ieeexplore.ieee.org/document/4781121/

[2] M. Saveski, "Cold Start Recommendations: A Non-negative Matrix Factorization Approach," p. 44.

[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009. [Online]. Available: http://ieeexplore.ieee.org/document/5197422/

[4] B. Yi, X. Shen, H. Liu, Z. Zhang, W. Zhang, S. Liu, and N. Xiong, "Deep Matrix Factorization with Implicit Feedback Embedding for Recommendation System," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8616805/inpress.

[5] C. R. Aberger, "Recommender: An Analysis of Collaborative Filtering Techniques," p. 5.

[6] "ACM RecSys challenge 2019 | Home." [Online]. Available: https://recsys.trivago.cloud/

[7] "TensorFlow." [Online]. Available: https://www.tensorflow.org/

[8] D. Zhang and W. S. Lee, "A Web-based Question Answering System," p. 5.

[9] S. Russel and P. Norvig, *Artificial intelligence : a modern approach*, 3rd ed. Prentice Hall, 2003.

[10] "compsci760-recommender-system," Jun. 2019, original-date: 2019-05-12T01:47:22Z. [Online]. Available: https://github.com/compsci760-recommender-system/DMF-model