Journal of Big Data

RESEARCH Open Access

Multi-criteria collaborative filtering recommender by fusing deep neural network and matrix factorization

Nour Nassar*, Assef Jafar and Yasser Rahhal

*Correspondence: nournassar123@gmail.com; nour.nassar@hiast.edu.sy
Higher Institute for Applied Sciences and Technology, Damascus, Syria

Abstract
Recommender systems have been an efficient strategy to deal with information overload by producing personalized predictions. Recommendation systems based on deep learning have accomplished magnificent results, but most of these systems are traditional recommender systems that use a single rating. In this work, we introduce a multi-criteria collaborative filtering recommender by combining deep neural network and matrix factorization. Our model consists of two parts: the first part uses a fused model of deep neural network and matrix factorization to predict the criteria ratings and the second one employs a deep neural network to predict the overall rating. The experimental results on two datasets, including a real-world dataset, show that the proposed model outperformed several state-of-the-art methods across different datasets and performance evaluation metrics.

Keywords: Collaborative filtering, Deep learning, Deep neural network, Matrix factorization, Multi-criteria, Recommender system

Introduction

In the era of big data, while the World Wide Web keeps growing exponentially, the size and complexity of many websites (Google, YouTube, Netflix, Amazon, and others) grow along with it, making it increasingly difficult and time-consuming for the users of these websites to find the item (movie, music, restaurant, book, and product in general) they are looking for. Recommender systems (RSs) provide personalized suggestion for items that the user might like [1]. Exploiting the information from users' ratings can be useful to solve one of the problems recommender systems suffer from, predicting users' preferences about an item using a single rating. This is a clear limitation, since the user who makes a choice might take into consideration more than one aspect of the item. For an example, in a movie recommender system, some users may like a movie based on its plot, direction, or conflict, while others may like the same movie but for its acting, characters, or any other attribute of that movie. So, Multi-Criteria Recommender System (MCRS) that utilizes multi-criteria ratings to evaluate different attributes of the item can improve the accuracy of the recommendations [2]. RSs use many techniques, Collaborative Filtering (CF) is the most commonly used, and it makes recommendations based only on interactions between users and items such as ratings [3]. Matrix Factorization (MF), is the most popular CF techniques, which maps users and items into a joint latent space, using a vector of latent features to represent a user or an item [4]. Then the interaction of a user on an item is obtained from the inner product of their latent vectors. Although MF is effective, the simple choice of the interaction function is insufficient to model the complex relation between the users and items.

Deep Learning (DL) has achieved immense results in many research fields, such as speech recognition, natural language processing, computer vision, and recently in recommender systems, where Deep Neural Networks (DNNs) have proved their capability of modeling nonlinear users and items relationships and approximating any continuous function [5].

Therefore, it is convenient to fuse the DNN with MF to formulate a more general model that makes use of both the nonlinearity of DNN and the linearity of MF to enhance recommendation accuracy.

The objective of this paper is just to propose a multi-criteria collaborative filtering recommender by fusing DNN and MF, to improve collaborative filtering performance. Our model consists of two parts: in the first part, we get the users and items' features and feed them as an input to a fused model of a DNN and MF to predict the criteria ratings, and we use a deep neural network to predict the overall rating in the second part.

By doing experiments on two datasets including a real-world dataset, it can be observed that our proposed model achieves significant improvement compared to the other state-of-the-art methods.

The main contributions of this work are as follows:

1. We present a multi-criteria collaborative filtering recommender by fusing DNN and MF, to combine the non-linearity of DNN and linearity of MF in a multi-criteria RS for modeling user-item latent structures.
2. We do comprehensive experiments on two datasets including a real-world dataset to show the efficiency of our model and the importance of using multi-criteria and deep learning in collaborative filtering.

The rest of this paper is organized as the following: In Sect. 2, we survey the related work. We will give a detailed overview of the system in Sect. 3. Section 4 presents the experimental evaluations and discussion and in Sect. 5, we provide a brief conclusion and potential future work.

Related work

Multi-criteria recommendation techniques can be divided into two general classes: memory-based and model-based methods. In memory-based methods, the similarity can be computed in two ways: the first approach calculates the similarity on each criteria rating separately using the traditional way and then aggregates the values into a single similarity using an aggregation method such as average [6], worst-case [6], and weighted sum of individual similarities [7]. The second approach uses the multidimensional distance metrics (Euclidean, Manhattan, and Chebyshev distance metrics) to calculate the distance between multi-criteria ratings [6]. Model-based methods use the user-item ratings to learn a predictive model and later use this model to predict unknown ratings, like Probabilistic Modeling [8, 9], Multilinear Singular Value Decomposition (MSVD) [10], Support Vector Regression (SVR) [11], and aggregation-function-based [6], this approach assumes that there is a relation between the item's overall rating and the multi-criteria ratings and it is not independent of other ratings.

Many efforts have been made to improve MF, Koren [12] merged MF and neighborhood models. Wang et al. [13] combined MF with topic models of item content. Rendle [14] introduced Factorization Machines that combine Support Vector Machines with factorization models. He et al. [15] proposed Neural Matrix Factorization (NeuMF) model that changed the linearity nature of MF by combining it with Multi-Layer Perceptron (MLP).

There are very few researches on applying deep learning to Collaborative Filtering model. Salakhutdinov et al. [16] proposed restricted boltzmann machines model to learn the item ratings correlation, thereafter Georgiev et al. [17] expanded the former work by adding both user–user and item–item correlations. Ouyang et al. [18] used autoencoder in autoencoder-based collaborative filtering to model users ratings on items. He et al. [15] introduced neural collaborative filtering model that uses MLP to learn the interaction function. Nassar et al. [19] presented deep multi-criteria collaborative filtering (DMCCF) model which is the only attempt in applying deep learning and multi-criteria to collaborative filtering. The model follows the aggregation-function-based approach, where they used a deep neural network to predict the criteria ratings and another DNN to learn the relationship between the criteria ratings and the overall rating in order to predict the overall rating.

Method

The proposed model is based on the model that Nassar et al. [19] presented. It contains three steps:

a. Predict criteria ratings $r_1, r_2, \ldots, r_k$ using a DNN.
b. Learn aggregation function f, which represents the relationship between the criteria ratings and the overall rating using a DNN.
c. Predict overall ratings using the predicted criteria ratings and the aggregation function f.

In our model, we used a fused model of DNN and MF to predict the criteria ratings $r_1, r_2, \ldots, r_k$ in the first step, while in the second step, we kept using a DNN to learn aggregation function f, as illustrated in Fig. 1.

Criteria ratings model

This model is used to predict the criteria ratings for a user on an item. In this model, we will fuse MF and DNN as He et al. [15] proposed in their NeuMF framework. MF that applies a linear kernel to model the latent feature interactions, and DNN that uses a non-linear kernel to learn the interaction function from data.

1. MF

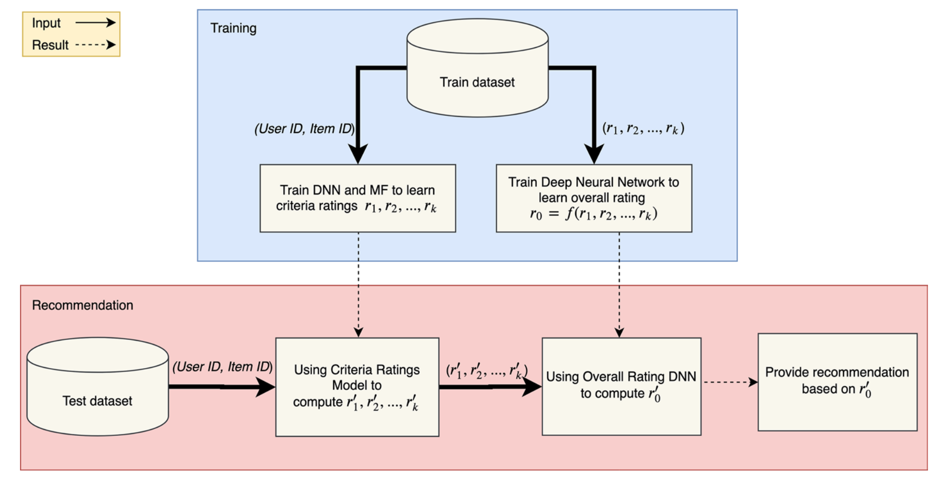
Input
Result
Training
Train dataset
(User ID, Item ID)
$(r_1, r_2, ..., r_k)$
Train DNN and MF to learn criteria ratings $r_1, r_2, ..., r_k$
Train Deep Neural Network to learn overall rating $r_0 = f(r_1, r_2, ..., r_k)$
Recommendation
Test dataset
(User ID, Item ID)
Using Criteria Ratings Model to compute $r'_1, r'_2, ..., r'_k$
$(r'_1, r'_2, ..., r'_k)$
Using Overall Rating DNN to compute r'_0
Provide recommendation based on r'_0

Fig. 1 The framework of the proposed model, including two steps: the Training and the Recommendation step

We can use the item ID or user ID as a feature since it is unique, but the ID is a categorical feature.

Therefore, we converted the IDs into embedding vectors initialized with random values that adjusted to minimize the loss function during the model training.

In the input layer, the input vector x is given by:

$$x = v_u \odot v_i \tag{1}$$

where v_u and v_i are user and item embedding vectors and $\odot$ is the element-wise product of vectors. The formula of the output layer is as follows:

$$y_{ui} = a_{out}\left(w^T (v_u \odot v_i)\right) \tag{2}$$

where a_{out} and w^T are the activation function and weights of the output layer.

2. DNN
The input vector x is given by:

$$x = Concatenate(v_u, v_i) = \begin{bmatrix} v_u \\ v_i \end{bmatrix} \tag{3}$$

where v_u and v_i are user and item embedding vectors.
Then followed by a number of dense Rectified Linear Units (ReLU) layers, we choose ReLU activation function because it is the most efficient [20]. The formulation of which is given as follows:

$$ReLU(z) = max(z, 0) \tag{4}$$

The output of a hidden layer l is formulated as:

$$h_l = ReLU\left(W_l h_{l-1} + b_l\right) \tag{5}$$

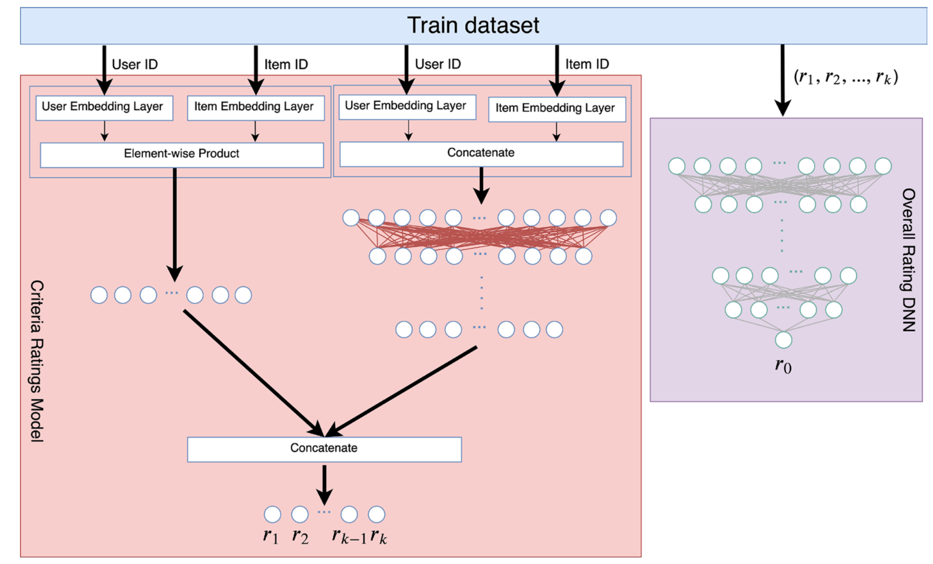

Fig. 2 The architecture of the two models: Criteria Ratings Model and Overall Rating DNN

where W_l, and b_l are the weight matrix and bias vector for the layer l and $h_1 = x$. In the output layer, we have:

$$y_{ui} = ReLU(W_L h_{L-1} + b_L) \tag{6}$$

where L is the number of layers.

3. Fusion of MF and DNN

For more flexibility to the fused model, we let DNN and MF learn separate embeddings, as shown in Fig. 2, and then we combine them by concatenating the last hidden layer. The outputs of the DNN and MF denoted as o^D, o^M respectively, are given as follows:

$$o^D = ReLU\left(W_L\left(ReLU\left(W_{L-1}\left(\ldots ReLU\left(W_2\begin{bmatrix} v_u^D \\ v_i^D \end{bmatrix} + b_2\right)\ldots\right) + b_{L-1}\right)\right) + b_L\right) \tag{7}$$

$$o^M = v_u^M \odot v_i^M \tag{8}$$

where v_u^D and v_u^M are user embedding vectors for DNN and MF part, and v_i^D and v_i^M are item embedding vectors.

In the model output layer, we predict the user criteria ratings $r_1, r_2, \ldots, r_k$ using Eq. (9) and (10):

$$y_{ui} = ReLU\left(W\begin{bmatrix} o^D \\ o^M \end{bmatrix} + b\right) \tag{9}$$

$$[r_1, r_2, \ldots, r_k]^T = y_{ui} \tag{10}$$

Overall rating deep neural network

The overall rating DNN is used to learn the aggregation function f, which represents the relationship between the overall rating r_0 and the criteria ratings $r_1, r_2, \ldots, r_k$, in order to predict the overall rating:

$$r_0 = f(r_1, r_2, \ldots, r_k) \tag{11}$$

In the input layer, the input vector is the criteria ratings $r_1, r_2, \ldots, r_k$ for user u and item i, as shown in Fig. 2. We normalize the continuous features $r_1, r_2, \ldots, r_k$ because the DNNs are sensitive to the inputs scaling and distribution [21]. The normalization of a sample r_i is calculated as:

$$z_i = \frac{r_i - m_i}{s_i} \tag{12}$$

with m_i the mean of the training samples for rating r_i and s_i the standard deviation of the training samples for rating r_i. The input vector becomes:

$$x = [z_1, z_2, \ldots, z_k]^T \tag{13}$$

Then followed by a number of hidden layers, the output of a hidden layer is given again by Eq. (**5**).

In the output layer, we predict the overall rating r_0 in Eq. (**6**), where:

$$r_0 = y_{ui} \tag{14}$$

We used Adam optimizer [22] and MAE loss function in both parts.

Recommendation

After the finishing of the model training, where each part was trained individually without knowing each other. We can use the model to predict the user's overall rating on the new items.

The recommendation process happens as shown in Fig. 1, for each user u and item i pair we:

a) Get the user ID and the item ID, and use them as an input for the Criteria Ratings model, to compute the criteria ratings $r'_1, r'_2, \ldots, r'_k$.
b) Normalize the previously computed criteria ratings $r'_1, r'_2, \ldots, r'_k$ and use them as an input for the Overall Rating DNN, to compute the overall rating r'_0.
c) Recommend the user the items as in traditional recommender systems using the overall rating r'_0.

Table 1 TripAdvisor dataset statistics

Field description	Value
Number of users	72,119
Number of hotels	1850
Number of ratings	81,085
Data sparsity	99.94%

Table 2 TripAdvisor dataset ratings distribution

Ratings	1	2	3	4	5
Value	4268	5853	10,374	24,696	35,894
Rooms	4002	5517	10,493	25,019	36,054
Location	1464	3141	8503	22,177	45,800
Cleanliness	2591	3585	8232	19,951	46,726
Check in/front desk	3775	4468	11,083	19,201	42,558
Service	4086	4462	10,036	20,114	42,387
Business service	4525	5849	19,624	21,661	29,426
Overall	4246	6052	8078	24,180	38,529

The number of examples of the values for each criteria rating and for the overall rating in the dataset

Table 3 Movies dataset statistics

Field description	Value
Number of users	6078
Number of movies	976
Number of ratings	62,156
Data sparsity	98.95%

The number of examples of the values for each criteria rating and for the overall rating in the dataset

Results and discussion

Dataset

We evaluated our model on two multi-criteria rating datasets, a real-world TripAdvisor[1] dataset and a Movies[2] dataset.

1. TripAdvisor dataset

   Multi-criteria rating dataset for hotels. It includes an overall rating and seven criteria ratings Value, Rooms, Location, Cleanliness, Check in/front desk, Service, and Business service, the rating range between 1 and 5. Table 1 demonstrates the statistics of the dataset and Table 2 the distribution of values for the different criteria ratings and the overall rating.
2. Movies dataset

[1] The dataset is crawled from the TripAdvisor website: https://www.tripadvisor.com/.

[2] https://github.com/an888ha/multi_criteria_recommender_system/blob/master/data_movies.txt.

Table 4 Movies dataset ratings distribution

Ratings	1	2	3	4	5	6	7	8	9	10	11	12	13
Criteria 1	3325	1046	1581	1151	1754	3018	2297	3462	6544	6582	5711	12,187	13,498
Criteria 2	2567	796	1133	861	1375	2729	2038	3475	7360	6743	6163	12,330	14,586
Criteria 3	3300	1067	1619	1113	1672	3073	2181	3477	7413	6328	5781	12,345	12,787
Criteria 4	2533	655	1063	758	1163	2654	1802	3135	7156	6365	5794	12,004	17,074
Overall	3395	1340	1522	1329	2051	2428	2489	3251	5586	7006	6702	12,153	12,904

Multi-criteria rating dataset for movies, available on GitHub. It contains four criteria ratings and an overall rating, the rating range between 1 and 13. Tables 3 and 4 demonstrate the dataset statistics and the distribution of the different criteria ratings and the overall rating.

Evaluation

To evaluate the performance of our model we used the same metrics used in [19].

a. Mean Absolute Error (MAE) [23]

$$MAE = \frac{1}{M}\sum_{u,i}|r_{ui} - \tilde{r}_{ui}| \tag{15}$$

where r_{ui} the true rating of user u for item i, $\tilde{r}_{ui}$ the predicted rating, and M is the size of the test set.

b. F-measure (F_1 and F_2) [23]

$$F_1 = \frac{2PR}{P+R} \tag{16}$$

$$F_2 = \frac{5PR}{4P+R} \tag{17}$$

where P the precision and R the recall.

c. Fraction of Concordant Pairs (FCP) [24]

$$n_c^u = \left|\left\{(i,j)|\tilde{r}_{ui}\rangle\tilde{r}_{uj} and r_{ui} > r_{uj}\right\}\right| \tag{18}$$

where n_c^u the number of concordant pairs for user u, and we calculate n_d^u the number of discordant pairs for user u in a similar way.

$$n_c = \sum_u n_c^u \tag{19}$$

$$n_d = \sum_u n_d^u \tag{20}$$

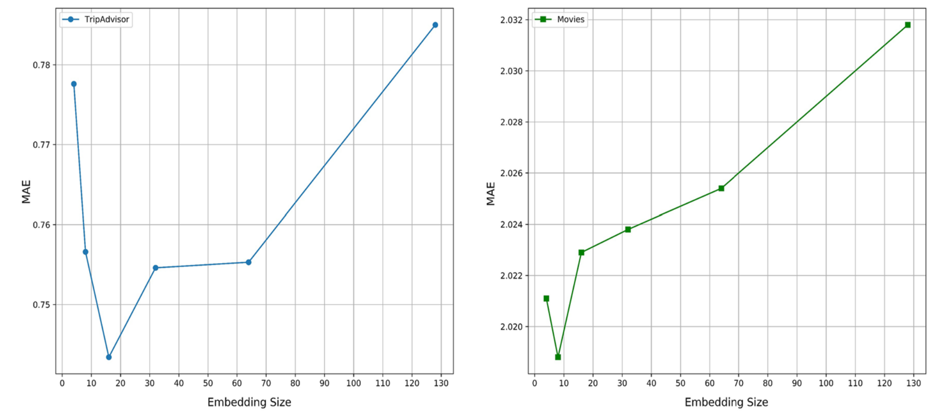

Fig. 3 The impact of embedding vector size in MF model for both datasets

$$FCP = \frac{n_c}{n_c + n_d} \tag{21}$$

d. Mean Average Precision (MAP) [23] is the average precision for all users.
e. Mean Reciprocal Rank (MRR) [25] is the average of the first relevant item rank for all users.

Settings

We used Keras[3] with TensorFlow[4] as a backend to implement our model.

1. Criteria Ratings Model Settings

We conducted several experiments to find the optimal parameters for DNN and MF.

- For DNN, we randomly initialized the DNN parameters using a normal distribution with mean of 0 and standard deviation of 0.05. We used Adam optimizer, with 0.001 learning rate and parameters values as provided in [22].

  For TripAdvisor dataset, we used batch size of 512 and set epochs to 2. We set each of user and item embedding vector size to 64, and we selected the $[128 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8]$ hidden layers. For Movies dataset, we used batch size of 64 and set epochs to 4. We set each of user and item embedding vector size to 32, and we selected the $[64 \rightarrow 32 \rightarrow 16 \rightarrow 8]$ hidden layers.
- For MF, we tried to find the optimal embedding vector size. As shown in Fig. 3, for TripAdvisor dataset, we set user and item embedding vector size to 16, and for Movies dataset, we set them to 8.
- In the model output layer, for TripAdvisor dataset, there are 7 neurons, equal to the number of the criteria ratings, and 4 neurons for Movies dataset.
- Overall Rating DNN Settings

[3] https://github.com/keras-team/keras.

[4] https://github.com/tensorflow/tensorflow.

Table 5 Single DNN settings

Dataset	Parameter	Value
TripAdvisor	User and item embedding vector size	256
	Hidden layers	[512 → 256 → 128 → 64 → 32 → 16 → 8]
	Output layer	1
	Batch size	512
Movies dataset	User and item embedding vector size	128
	Hidden layers	[256 → 128 → 64 → 32 → 16 → 8]
	Output layer	1
	Batch size	64

We initialized randomly the DNN parameters like the previous DNN, using a normal distribution with mean of 0 and standard deviation of 0.05. We also used Adam optimizer with 0.001 learning rate and the same parameters values. For TripAdvisor dataset, we set the epochs to 50, and for Movies dataset, we set the epochs to 100. While for both datasets, we set the batch size to 512. We used [64 → 32 → 16 → 8] hidden layers. Finally, in the output layer, there is 1 neuron, for the overall rating.

Results

We used fivefold cross-validation method, to split the data randomly into 20% test set and 80% training set including a 1% validation set. We repeated this process 5 times and calculated the mean value of each metric. We compared the performance of our model to the DMCCF model [19], then to a single DNN that predicts the overall rating directly where the finest results of this DNN were acquired from settings shown in Table 5. In addition, we compared our model to a number of famous single rating recommendation methods such as SVD [26], SVD++ [12], Baseline Estimates [27], and SlopeOne [28].

This comparison was done on the overall rating. The results are illustrated in Table 6. We can see that our model achieves the best performance on both datasets, significantly outperforming each of DMCCF model, single DNN, and the other state-of-the-art methods on all the evaluation metrics. This indicates the high expressiveness of our model by fusing the non-linear DNN and linear MF models to capture the methods. F1 and F2 of our model are better than the other methods. Our model surpasses the other models in FCP and it exceeds user-item interactions. According to the results, in MAE, our model excelled the other compared methods at MAP. In MRR, our model is the best.

Conclusion and future work

In this paper, we proposed multi-criteria collaborative filtering recommender by fusing DNN and MF. The model consists of two parts: in the first part, we get the users and items' features and feed them as an input to a fused model of a DNN and MF to predict the criteria ratings, and in the second part, we use a DNN to predict the overall rating. Then we demonstrated the efficiency of our proposed model, where the experimental results showed that our model significantly outperformed the other methods on both datasets for all the evaluation metrics, which proved that the application of deep

Table 6 Evaluation results

Dataset	Metric	SVD	SVD++	SlopeOne	Baseline estimates	Single DNN	DMCCF [19]	Our
TripAdvisor	*MAE*	0.8071 ± 0.0043	0.8222 ± 0.0071	0.9160 ± 0.0053	0.8094 ± 0.0019	0.7855 ± 0.0153	0.7552 ± 0.0050	0.7434 ± 0.0068
	F_1	0.8370 ± 0.0035	0.8390 ± 0.0035	0.8666 ± 0.0018	0.8391 ± 0.0020	0.8572 ± 0.0024	0.8769 ± 0.0043	0.8811 ± 0.0031
	F_2	0.7951 ± 0.0031	0.7991 ± 0.0057	0.9248 ± 0.0011	0.7986 ± 0.0029	0.9176 ± 0.0057	0.9295 ± 0.0014	0.9319 ± 0.0204
	FCP	0.5024 ± 0.0210	0.5092 ± 0.0056	0.5096 ± 0.0661	0.5025 ± 0.0117	0.4957 ± 0.0102	0.5126 ± 0.0160	0.5164 ± 0.0048
	MAP	0.3935 ± 0.0057	0.3483 ± 0.0049	0.3006 ± 0.0026	0.3881 ± 0.0023	0.4481 ± 0.0091	0.4643 ± 0.0072	0.4836 ± 0.0047
	MRR	0.7272 ± 0.0054	0.7513 ± 0.0018	0.7502 ± 0.0037	0.7374 ± 0.0028	0.7244 ± 0.0092	0.7664 ± 0.0039	0.7693 ± 0.0036
Movies dataset	*MAE*	2.1072 ± 0.0168	2.1594 ± 0.0050	2.1798 ± 0.0112	2.1333 ± 0.0069	2.0613 ± 0.0347	2.0474 ± 0.0119	2.0188 ± 0.0181
	F_1	0.8061 ± 0.0049	0.7985 ± 0.0010	0.7860 ± 0.0026	0.8107 ± 0.0031	0.8344 ± 0.0106	0.8414 ± 0.0041	0.8563 ± 0.0036
	F_2	0.7608 ± 0.0062	0.7553 ± 0.0023	0.7325 ± 0.0045	0.7699 ± 0.0045	0.8256 ± 0.0287	0.8383 ± 0.0049	0.8442 ± 0.0056
	FCP	0.6022 ± 0.0107	0.5985 ± 0.0029	0.6251 ± 0.0098	0.6233 ± 0.0027	0.6226 ± 0.0055	0.6258 ± 0.0083	0.6289 ± 0.0029
	MAP	0.1910 ± 0.0056	0.2144 ± 0.0035	0.2089 ± 0.0067	0.1612 ± 0.0027	0.2136 ± 0.0143	0.2345 ± 0.0109	0.2458 ± 0.0090
	MRR	0.6651 ± 0.0091	0.6502 ± 0.0048	0.6085 ± 0.0048	0.6721 ± 0.0052	0.6922 ± 0.0145	0.7061 ± 0.0078	0.7092 ± 0.0049

Evaluation results for our model against the other algorithms. F_1, F_2, *FCP*, *MAP*, and *MRR* the higher the better while *MAE* the lower the better learning and multi-criteria in collaborative filtering is a successful attempt and it can be enhanced using different deep learning techniques or by building more complex models.

In future work, we will study the use of different deep learning techniques, such as Autoencoder, Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN) in recommendation systems and attempt to further improve the performance of our model, we will also try other feature representation methods precisely to solve the cold start problem by using user and item content features.

Abbreviations

CF: Collaborative filtering; DL: Deep learning; DMCCF: Deep multi-criteria collaborative filtering; DNN: Deep neural network; FCP: Fraction of Concordant Pairs; MAE: Mean Absolute Error; MAP: Mean Average Precision; MCRS: Multi-Criteria Recommender System; MF: Matrix factorization; MLP: Multi-layer perceptron; MRR: Mean Reciprocal Rank; NeuMF: Neural Matrix Factorization; ReLU: Rectified Linear Unit; RS: Recommender system.

Acknowledgements

The authors thank N. Mohanna for careful reading of the manuscript. Special thanks are due to S. Tarbouche for advice and support.

Authors' contributions

All the corresponding authors contributed equally to the conduct of the present study. All authors read and approved the final manuscript.

Funding

Not applicable.

Availability of data and materials

The datasets used during the current study are available in the Datasets repository, https://github.com/nournassar/Datasets.

Competing interests

The authors declare that they have no competing interests.

Received: 18 October 2019 Accepted: 18 May 2020
Published online: 24 May 2020

References

1. Zhao J, Geng X, Zhou J, Sun Q, Xiao Y, Zhang Z, Fu Z. Attribute mapping and autoencoder neural network based matrix factorization initialization for recommendation systems. Knowl Based Syst. 2019;166:132–9.
2. Turk AM, Bilge A. Robustness analysis of multi-criteria collaborative filtering algorithms against shilling attacks. Expert Syst Appl. 2019;115:386–402.
3. Zhang S, Yao L, Sun A, Tay Y. Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv. 2019;52:5.
4. Fu M, Qu H, Yi Z, Lu L, Liu Y. A novel deep learning-based collaborative filtering model for recommendation system. IEEE Trans. Cybern. 2018. https://doi.org/10.1109/tcyb.2018.2795041.
5. Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. Neural Networks. 1989;2:359–66.
6. Adomavicius G, Kwon Y. New recommendation techniques for multicriteria rating systems. IEEE Intell Syst. 2007;22:48–55. https://doi.org/10.1109/MIS.2007.58.
7. Tang TY, Mccalla G. The pedagogical value of papers: a collaborative-filtering based paper recommender. J. Digit. Inf. 2009. https://doi.org/10.1109/wkdd.2009.132.
8. L. Si, R. Jin, Flexible Mixture Model for Collaborative Filtering. In: Proc. Twent. Int. Conf. Int. Conf. Mach. Learn., AAAI Press, 2003: pp. 704–711. http://dl.acm.org/citation.cfm?id=3041838.3041927.
9. Sahoo N, Krishnan R, Duncan G, Callan J. The halo effect in multicomponent ratings and its implications for recommender systems: the case of yahoo! movies. Inf. Syst. Res. 2012;23:231–46. https://doi.org/10.1287/isre.1100.0336.
10. Q. Li, C. Wang, G. Geng, Improving personalized services in mobile commerce by a novel multicriteria rating approach. In: Proceeding 17th Int. Conf. World Wide Web - WWW'08. (2008) 1235. https://doi.org/10.1145/1367497.1367743.
11. Drucker H, Burges CJC, Kaufman L, Smola A, Vapnik VN. Support vector regression machines. Adv Neural Inf Process Syst. 1997;9(1):155–61.
12. Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., ACM, New York, NY, USA, 2008: pp. 426–34. https://doi.org/10.1145/1401890.1401944.
13. Wang H, Wang N, Yeung DY. Collaborative deep learning for recommender systems. In: Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., 2015. pp. 1235–44.
14. Rendle S. Factorization machines. In: Data Min. (ICDM), 2010 IEEE 10th Int. Conf., 2010: pp. 995–1000.

15. He X, Liao L, Zhang H, Nie L, Hu X, Chua T-S. Neural Collab Filter. 2017. https://doi.org/10.1145/3038912.3052569.
16. Salakhutdinov R, Mnih A, Hinton G. Restricted boltzmann machines for collaborative filtering. In: Proc. 24th Int. Conf. Mach. Learn., ACM, New York, NY, USA, 2007: pp. 791–98. https://doi.org/10.1145/1273496.1273596.
17. Georgiev K, Nakov P. A non-IID framework for collaborative filtering with restricted boltzmann machines. In: S. Dasgupta, D. McAllester (Eds.), Proc. 30th Int. Conf. Mach. Learn., PMLR, Atlanta, Georgia, USA, 2013. pp. 1148–56. http://proceedings.mlr.press/v28/georgiev13.html.
18. Ouyang Y, Liu W, Rong W, Xiong Z. Autoencoder-based collaborative filtering. In: Loo CK, Yap KS, Wong KW, Beng AT, Huang K, editors. Neural Inf. Cham: Process. Springer International Publishing; 2014. p. 284–91.
19. Nassar N, Jafar A, Rahhal Y. A novel deep multi-criteria collaborative filtering model for recommendation system. Knowledge-Based Syst. 2019. https://doi.org/10.1016/j.knosys.2019.06.019.
20. Fuentes O, Parra J, Anthony EY, Kreinovich V. Why rectified linear neurons are efficient: symmetry-based, complexity-based, and fuzzy-based explanations. Dep. Tech. Reports (CS). (2017). https://digitalcommons.utep.edu/cs_techrep/1171.
21. Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv. 2015. https://doi.org/10.1007/s13398-014-0173-7.2.
22. D.P. Kingma, J. Ba. Adam: A Method for Stochastic Optimization. In *ICLR*: International Conference on Learning Representations. 2014. .
23. Herlocker JL, Konstan JA, Terveen LG, Riedl JT. Evaluating collaborative filtering recommender systems. ACM Trans Inf Syst. 2004;22:5–53.
24. Koren Y, Sill J. Collaborative filtering on ordinal user feedback. In: IJCAI Int. Jt. Conf. Artif. Intell., 2013: pp. 3022–26. https://doi.org/10.1145/1985793.1986027.
25. Abel F, Araújo S, Gao Q, Houben GJ. Analyzing cross-system user modeling on the social web. In: Int. Conf. Web Eng., 2011: pp. 28–43.
26. Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. 2009, p. 42–49. https://doi.org/10.1109/mc.2009.263.
27. Koren Y. Factor in the neighbors. ACM Trans Knowl Discov Data. 2010;4:1–24. https://doi.org/10.1145/1644873.1644874.
28. Lemire D, Maclachlan A. Slope one predictors for online rating-based collaborative filtering. 2007. doi: 10.1137/1.9781611972757.43

Publisher's Note