An Improved Hybrid Recommender System: Integrating Document Context-Based and Behavior-Based Methods

Meysam Varasteh  
*Electrical and Computer Engineering*  
University of Tehran  
Tehran, Iran  
meysamvaraste@ut.ac.ir

Mehdi Soleiman Nejad  
*Electrical and Computer Engineering*  
University of Tehran  
Tehran, Iran  
m.soleimannejad@ut.ac.ir

Hadi Moradi  
*Electrical and Computer Engineering*  
University of Tehran  
Tehran, Iran  
moradih@ut.ac.ir

Mohammad Amin Sadeghi  
*Electrical and Computer Engineering*  
University of Tehran  
Tehran, Iran  
asadeghi@ut.ac.ir

Ahmad Kalhor  
*Electrical and Computer Engineering*  
University of Tehran  
Tehran, Iran  
akalhor@ut.ac.ir

Abstract—One of the main challenges in recommender systems is data sparsity, which leads to high variance. Several attempts have been made to improve the bias-variance trade-off using auxiliary information. In particular, document modeling-based methods have improved the model's accuracy by using textual data such as reviews, abstracts, and storylines when the user-to-item rating matrix is sparse. However, such models are insufficient to learn optimal representation for users and items. For building recommender systems, user-based and item-based collaborative filtering have long been used due to their efficiency. A user and item profile are created based on their historically interacted items and the users who interacted with the target item. In spite of the fact that these two approaches have been studied separately, there has been little research into combining them.

The purpose of this study is to combine these two approaches by considering the opinions of users on these items. Each user is represented by their historical behavior, while each item is represented by the users who have interacted with it before, combined with contextual information, which is processed with NLP. The proposed algorithm is implemented and tested on three real-world datasets that demonstrate our model's effectiveness over the baseline methods.

Index Terms—Recommender Systems, Matrix Factorization, CNN, User Modeling, Contextual Information

I. INTRODUCTION

Nowadays, recommender systems play an integral role in our lives; with the ever-increasing amount of information, these systems guide many aspects of our lives by providing important and relevant cases and filtering non-relevant ones. One of the main objectives of recommender systems is to model user preferences for items based on the recorded information [1]. User preferences can be extracted through their ratings, clicks, or percentage of views. In most online services, customers can submit their reviews for products and share their opinions with other customers to help them. Researches show that nearly one-third of online shoppers refuse to purchase products that have not received positive feedback from customers; Therefore, it is a mutual benefit for users and the company and simultaneously increases user satisfaction and corporate profit [2, 3, 4]. Researchers have recently been using this valuable information to represent users and items better and handle the sparsity problem [5]. Various document-based modeling approaches such as Latent Dirichlet Allocation (LDA) and Stacked Denoising Auto-Encoder (SDAE) have been proposed to improve the accuracy of Recommender Systems by utilizing textual data such as reviews and storylines [5, 6, 7]. In addition, some methods have been proposed by integrating the aforementioned topic modeling methods with Collaborative Filtering, known as Collaborative Topic Regression (CTR) [6, 8]. However, such integrated approaches do not fully capture document information. To address this issue, some works like [9, 10, 11] utilize Convolutional Neural Networks (CNN). CNNs facilitate a deeper understanding of documents and generate a better latent vector than topic modeling methods. To address the two limitations mentioned above, we aim to build a Hybrid RecSys. We create a profile for each user and characterize the user with their interaction history, considering opinion information rather than their ID. These opinions of users on items can capture users’ preferences on items that provide better representation. Likewise, We build a profile for each item based on the set of users who have interacted with the target item; even users might express different opinions about each item. By combining these pieces of information with the textual description, we can capture the characteristics of the same item from different perspectives.
II. RELATED WORK

In this section, we briefly review several closely related works, including general recommendation methods and context-aware recommender systems.

A. General Recommendation

Collaborative filtering (CF) is a popular and widely used filtering method based on the idea that our future behavior is influenced by our past behavior [1, 12]. We often get the best recommendations from other users who have similar tastes to ours [13]. Matrix Factorization (MF) is a class of the CF algorithm. The idea behind MF is to represent each user and item in a lower dimension latent space, and the objective is to exploit the relationship between users and items in latent vectors [1, 14, 15]. By multiplying these vectors, a user’s preference, \( U \), for an item, \( I \), is obtained. For example, if we have \( N \) users, \( M \) items and rating matrix \( R \subset \mathbb{R}^{N \times M} \), the user and item latent vector respectively are \( u_i \subset \mathbb{R}^{k} \) and \( v_j \subset \mathbb{R}^{k} \) where \( k \) is the latent space size. The predicted rating of \( U \) to \( I \) is calculated by multiplying the corresponding vectors. In recent years, deep learning, due to its capability of approximating any continuous function and capturing intricate patterns, has garnered attention in many fields, including recommender systems[16, 17]. One line of deep learning-based methods in recommendation systems is used to estimate user preferences. For example, [18] fused CF with neural architecture, which considers both the linearity of MF and the non-linearity of neural architecture to enhance ranking performance. Another line of work uses auxiliary information such as text, image, and acoustic features in the CF model, which we elaborate on in the following subsection.

B. Context-Aware Recommender Systems

A context-aware recommender system (CARS) uses contextual information to model users’ preferences during the recommendation process. Research has shown that contextual information can improve the accuracy of recommendations and alleviate the sparsity issue [10, 11]. One of the most important and widely used tools in text processing is CNN which has become the cornerstone of deep learning [9, 10, 11]. Taking advantage of this success, [11] uses CNN architecture to capture contextual information of the movie’s description and combines it with probabilistic matrix factorization (PMF) to enhance rating prediction [19]. In recent years, another stunning success in deep learning was the attention mechanism, which will be discussed. [9, 10] has used this mechanism and integrated it with residual networks. [20, 21] by utilizing conditional RNNs, proposed a context-aware session-based RecSys, which injects contextual information into input and output layers. It modifies the behavior of the RNN by combining context embedding with item embedding. [21] to address the sparsity problem and improve the accuracy of the CARS, proposed a model which uses the encoding-decoding process to learn latent context representations and utilize sequences of user data derived from contextual conditions.
III. OUR MODEL

A. Problem Statement

Let $U = \{u_1, u_2, ..., u_n\}$ and $V = \{v_1, v_2, ..., v_m\}$ be the sets of users and items respectively where $n$ is the number of users and $m$ is the number of items. In recommender systems, the user-item rating matrix is denoted by $R \in \mathbb{R}^{n \times m}$ where each row represents a user $u_j$ with $1 \leq j \leq n$ and each column represents an item $i_k$ with $1 \leq k \leq m$. The elements of this matrix are the ratings that are given to items by users. An example input of our model is illustrated as follows:

$$\begin{bmatrix}
[u_1, u_2, ..., u_{n'}] \\
[w_1, w_2, ..., w_{k'}] \\
[v_1, v_2, ..., v_{m'}]
\end{bmatrix}$$

Where $n' < n$, $k' < k$ and $m' < m$. Given the above inputs, our model aims to predict the rating.

B. User Section

The user section aims to learn the user latent vector by using past interactions and opinions of the user. Each user is mapped to a multi-hot vector composed of movies that the user has rated. A user can express their satisfaction, denoted as a rating score, which means that all interactions are not of equal importance. We use this information and, similar to [22], combine the rating embedding with the user embedding. By passing each input vector through an attention network, the attention weights are obtained. Next, each vector is multiplied by the corresponding weight. Finally, the vector is passed through a fully connected layer.

C. Item Section

As shown in Fig 1, the item section consists of two parts:

Document Information Modeling Part: We use the popular pre-trained word embedding model, GloVe \(^1\), that converts each word to a dense vector with a fixed length. Suppose we have $\rho$ words in a document that describes the item, so we obtain a matrix $D \in \mathbb{R}^{\rho \times l}$ where $l$ is the size of embedding for each word. As shown in Fig 2, we use one dimensional CNN with multiple filters to project input to vectors and capture various types of contextual features:

$$c^j_i = f(W.x(i + i + h - 1) + b)$$

Where $c^j_i \in \mathbb{R}$, $i$ is the number of convolution operations, $m$ is the number of convolutional kernels, $f$ is the activation function, $W$ is the weights, and $b \in \mathbb{R}$ is the bias. As to the variable length of each document, we use the convolution architecture in [23]. After passing the document through the convolution layer, we obtain a feature vector with variable length for each kernel weight. By using max-pooling, each vector is converted to a scalar that extracts the features from the previous layer.

$$q = \left[\max(c^1), \max(c^2), ..., \max(c^j), ..., \max(c^m)\right]$$

Where $q \in \mathbb{R}^n$ is a fixed-length vector. Finally, we pass the vector through a fully connected layer.

User Information Modeling Part: Like the user modeling section, we represent each item as a set of users interacting with the target item. As we said, each user can express their satisfaction, denoted as rating score, which means that all interactions are not of equal importance. We use this information and, similar to [22], combine the rating embedding with the user embedding. By passing each input vector through an attention network, the attention weights are obtained. Next, each vector is multiplied by the corresponding weight. Finally,

\(^1\)https://nlp.stanford.edu/projects/glove/
we concatenate these two vectors and pass them through a fully connected layer that matches the user’s latent vector with this item’s one.

D. Rating Prediction Section

Finally, the user and item latent vectors are obtained according to the method proposed in the previous sections. Then, they are concatenated and passed through fully connected layers to predict ratings using the following equations:

\[ L_1 = \sigma_1(W_1 \times \text{Concat}(U, I) + b_1) \]
\[ L_2 = \sigma_2(W_2 \times L_1 + b_2) \]
\[ \vdots \]
\[ L_k = \sigma_k(W_k \times L_{k-1} + b_k) \]

In which \( k \), \( \sigma_i \), \( W_i \), and \( b \) respectively denote the number of layers, activation function, weights matrix, bias vector. In addition, \( U \) and \( I \) represent user and item latent vectors.

IV. EXPERIMENT

In this section we first introduce the evaluation datasets and evaluation methods. Then we analyze our model from different aspects.

A. Experimental Settings

1) Datasets: To demonstrate the effectiveness of our model, we used three available datasets, Movielens[24]^2 and Amazon^3. The datasets contain user ratings of items on a scale of 1 to 5 as Table I.

| Dataset | # user | # item | # Interaction | Density |
|---------|--------|--------|--------------|---------|
| ML-1m  | 6,040  | 3,544  | 993,482      | 4.641 % |
| ML-10m | 69,878 | 10,073 | 9,945,875    | 1.413 % |
| Amazon | 29,757 | 15,149 | 135,188      | 0.030 % |

Since our evaluation datasets do not contain the item’s description, we used IMDB^4 dataset, which includes the storyline and summary of the movies that was provided by MovieLens and Amazon researchers.

2) Optimization and Evaluation Metric: To evaluate our model and compare with the baselines, we use Root Mean Squared Error (RMSE), Hit Ratio, and Normalized Discounted Cumulative Gain (NDCG) as evaluation metrics that are defined as:

\[ RMSE = \sqrt{\frac{\sum_{i,j=1}^{N,M} (r_{ij} - \hat{r}_{ij})^2}{T}} \]  
\[ HR@K = \frac{\# \text{ cache hits}}{\# \text{ cache hits} + \# \text{ cache misses}} \]  
\[ NDCG@K = Z_k \sum_{i=1}^{K} \frac{2^{r_i} - 1}{\log_2(i+1)} \]

Where \( N, M, \hat{r}_{ij} \) and \( T \) respectively denote the number of the users, items, estimated value of the rating, and the total number of the ratings. Also, \( Z_K \) is the normalizer to ensure the perfect ranking has a value of 1; \( r_i \) is the graded relevance of item at position \( i \) [25].

3) Baselines: We compared our model with the following baselines:

- PMF [19]: Probabilistic Matrix Factorization is a standard rating prediction model which only utilizes user-item rating matrix and models latent factors of users and items by gaussian distribution.
- CTR [7]: Collaborative Topic Regression is a state-of-the-art model which combines PMF and Latent Dirichlet Allocation (LDA) to predict rating of user \( u \) on item \( i \).
- CDL [6]: Collaborative Deep Learning is a model that combines auto-encoders and PMF which analyze documents by SDAE.
- Conv MF [11]: Convolutional Matrix Factorization is a recent model that uses document information for items as input and combines PMF and CNN methods to predict rating.
- NeuMF [18]: NeuMF is a state-of-the-art Matrix Factorization model without document information. The original implementation is for ranking task and we adjust its loss to square loss for rating prediction.
- Att-ConvCF [9]: Attentional ConvCF is a document context aware model that integrates attention mechanism with CF to enhance rating prediction.
- GNNSR^5 [22]: GNNSR uses graph neural network framework (GraphRec) for social recommendations.
- RConvCF [10]: Residual ConvCF applies residual idea to word embeddings in order to capture semantic information and solve the gradient vanishing problem.

B. Results

This subsection studies the impact of the rating information, attention mechanism, effect of user information, and word embedding size in item modeling.

1) Effect of the Rating Information: A user can express their opinion of the items, which means that all interactions do not have the same importance for the target user. If a user likes the target item very much, they will give it a high score. In this subsection, we compare these two cases and show the results as Fig 3a. It is essential to point out that we didn’t consider user information in item modeling for all cases in this subsection. As shown in Fig3a if we consider the rating information in our model, it achieves better performance.

2) Effect of the Attention Mechanism: To better understand the proposed model and effectiveness of the involved attention mechanism, we replace it in user and item modeling with Max-Pooling and Mean-Pooling. Fig3b shows the results on ML-1m. Like the previous subsection, we didn’t consider the user information in item modeling. As shown in Fig3b, The results have met our previous expectation, which means that

^2https://movielens.org/
^3http://jmcauley.ucsd.edu/data/amazon/
^4https://www.imdb.com/
^5Graph Neural Networks for Social Recommendation.
we have the best performance if we consider attention in our model, the Mean-Pooling achieves better performance than Max-Pooling.

3) **Effect of the User Information in the Item Modeling:**
We now focus on analyzing the effectiveness of the user information in item modeling. At first, we consider the item's description as input of the item section and ignore the user information. Then in the following case, we simultaneously consider the user information and item's description as item input which the results are given in Fig 3c. The results show that combining users’ behavior and textual information can improve the performance of the model.

4) **Effects of the Document Information in the Item Modeling:**
In Fig 3e, We can see that without document information modeling, the performance of rating prediction has deteriorated significantly, and it justifies our assumption that document information modeling on item section has information that can help the model to learn each item’s latent vector and improve the recommendation performance.

5) **Effects of the Embedding Size in the Document Information:**
To better understand the impact of word embedding size, we analyzed the effect of word embedding size on the performance. Figure 3f shows the performance of our models with respect to various word embedding sizes. When we increase embedding size from 100 to 200, the RMSE does not boost; setting the value to 300 leads to the best result.

6) **Effects of the Word Embedding Pre-trained Model:**
In Fig 3d, we investigate the impact of the pre-trained word embedding models such as GloVe on our task by initializing GloVe’s pre-trained word embedding model.

7) **Performance Comparison:**
Table II summarizes the results of all models on three benchmark datasets, where underlined scores are the best competitor result and bold scores are our model’s results. The last row shows the improvement of our model to the best baseline. The results for another objective are not available. For example, PMF optimizes just rating prediction errors, so the HR and NDCG metrics results are not available. The original implementation of ConvMf, Att-ConvCf, GNNSR, and xRConvCF is for the rating prediction recommendation task; we adjust their loss to binary cross-entropy loss for ranking purposes. Also, the original implementations of NCF is for the ranking recommendation task, which we adapt its loss to the rating prediction recommendation task.

Table II shows our model’s results, overall rating prediction error (RMSE), HR, and NDCG, over ten baselines on three datasets, which demonstrates the effectiveness of our model over the state-of-the-art competitors.

**V. CONCLUSION**

This paper aimed to handle the sparsity problem and improve recommendation accuracy by considering contextual information and combining it with historical data. In the user section, we built a profile for each user based on their interacted items, and similarly, we created a profile for each item based on the users who have interacted with it. We integrated it with contextual information, which we processed with the CNNs approach. We analyzed the effect of each section on rating prediction and evaluated our model on three real-world datasets, which demonstrated the effectiveness of our model over the state-of-the-art competitors.
In future work, using one of the state-of-the-art models such as NeuMF, we try to find friends for each user according to the user embedding vector. We then incorporate the information of these trusted friends into the user modeling process. Another future work would be exploiting bidirectional models such as BERT[26] to provide a representation for each word by jointly conditioning on both left and right contexts.

REFERENCES

[1] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009.

[2] G. Adomavicius and A. Tuzhilin, “Context-aware recommender systems,” in Recommender systems handbook. Springer, 2011, pp. 217–253.

[3] S. Utz, P. Kerkhof, and J. Van Den Bos, “Consumers rule: How consumer reviews influence perceived trustworthiness of online stores,” Electronic Commerce Research and Applications, vol. 11, no. 1, pp. 49–58, 2012.

[4] B. Von Helversen, K. Abramczuk, W. Kopć, and R. Nielek, “Influence of consumer reviews on online purchasing decisions in older and younger adults,” Decision Support Systems, vol. 113, pp. 1–10, 2018.

[5] J. McAuley and J. Leskovec, “Hidden factors and hidden topics: understanding rating dimensions with review text,” in Proceedings of the 7th ACM conference on Recommender systems, 2013, pp. 165–172.

[6] H. Wang, N. Wang, and D.-Y. Yeung, “Collaborative deep learning for recommender systems,” in Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 1235–1244.

[7] C. Wang and D. M. Blei, “Collaborative topic modeling for recommending scientific articles,” in Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, 2011, pp. 448–456.

[8] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix
TABLE II: Performance Comparison of different methods on ML-1m, ML-10m, and Amazon datasets. Bold Scores are the best, while underlines scores are the second best. Some of the elements are "N/A", which means that the baseline model has optimized just one objective function.

| Baselines | ML-1m HR@10 | NDCG@10 | RMSE | ML-10m HR@10 | NDCG@10 | RMSE | Amazon HR@10 | NDCG@10 | RMSE |
|-----------|-------------|---------|------|-------------|---------|------|-------------|---------|------|
| PMF       | N/A         | N/A     | 0.897| N/A         | N/A     | 0.831| N/A         | N/A     | 1.412|
| POP       | 0.136       | 0.062   | N/A  | 0.129       | 0.04    | N/A  | 0.054       | 0.032   | NA   |
| CTR       | N/A         | N/A     | 0.897| N/A         | N/A     | 0.828| N/A         | N/A     | 1.550|
| BPR-MF    | 0.430       | 0.237   | N/A  | 0.342       | 0.170   | N/A  | 0.216       | 0.099   | N/A  |
| CDL       | N/A         | N/A     | 0.888| N/A         | N/A     | 0.819| N/A         | N/A     | 1.359|
| NCF       | 0.348       | 0.164   | 0.874| 0.301       | 0.135   | 0.898| 0.265       | 0.070   | 1.24 |
| ConvMF    | 0.497       | 0.296   | 0.855| 0.496       | 0.295   | 0.793| 0.357       | 0.236   | 1.128|
| Att-ConvCF| 0.501       | 0.301   | 0.740| 0.493       | 0.287   | 0.760| 0.398       | 0.251   | 0.772|
| GNNSR     | 0.510       | 0.306   | 0.751| 0.501       | 0.294   | 0.742| 0.385       | 0.236   | 0.783|
| xRConvCF  | 0.501       | 0.292   | 0.737| 0.513       | 0.293   | 0.746| 0.368       | 0.215   | 0.752|
| Our-Model | **0.558**   | **0.342**| **0.645**| **0.540**   | **0.339**| **0.632**| **0.404**   | **0.261**| **0.581**|

Improvement | 9.29% | 11.77% | 12.41% | 5.34% | 15.4% | 14.8% | 4.9% | 10.7% | 22.8% |

factorization,” in Advances in neural information processing systems, 2008, pp. 1257–1264.
[9] B. Zhang, H. Zhang, X. Sun, G. Feng, and C. He, “Integrating an attention mechanism and convolution collaborative filtering for document context-aware rating prediction,” IEEE Access, vol. 7, pp. 3826–3835, 2018.
[10] B. Zhang, M. Zhu, M. Yu, D. Pu, and G. Feng, “Extreme residual connected convolution-based collaborative filtering for document context-aware rating prediction,” IEEE Access, vol. 8, pp. 53604–53613, 2020.
[11] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, “Convolutional matrix factorization for document context-aware recommendation,” in Proceedings of the 10th ACM conference on recommender systems, 2016, pp. 233–240.
[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in Proceedings of the 10th international conference on World Wide Web, 2001, pp. 285–295.
[13] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Isip et al., “Wide & deep learning for recommender systems,” in Proceedings of the 1st workshop on deep learning for recommender systems, 2016, pp. 7–10.
[14] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, “Collaborative filtering and deep learning based recommendation system for cold start items,” Expert Systems with Applications, vol. 69, pp. 29–39, 2017.
[15] M. Deshpande and G. Karypis, “Item-based top-n recommendation algorithms,” ACM Transactions on Information Systems (TOIS), vol. 22, no. 1, pp. 143–177, 2004.
[16] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” ACM Computing Surveys (CSUR), vol. 52, no. 1, pp. 1–38, 2019.
[17] R. Mu, “A survey of recommender systems based on deep learning,” IEEE Access, vol. 6, pp. 69009–69022, 2018.
[18] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in Proceedings of the 26th international conference on world wide web, 2017, pp. 173–182.
[19] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” Advances in neural information processing systems, vol. 20, pp. 1257–1264, 2007.
[20] E. Smirnova and F. Vasile, “Contextual sequence modeling for recommendation with recurrent neural networks,” in Proceedings of the 2nd workshop on deep learning for recommender systems, 2017, pp. 2–9.
[21] A. Livne, M. Unger, B. Shapira, and L. Rokach, “Deep context-aware recommender system utilizing sequential latent context,” arXiv preprint arXiv:1909.03999, 2019.
[22] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, “Graph neural networks for social recommendation,” in The World Wide Web Conference, 2019, pp. 417–426.
[23] A. Rakhlin, “Convolutional neural networks for sentence classification,” GitHub, 2016.
[24] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” Acm transactions on interactive intelligent systems (tiis), vol. 5, no. 4, pp. 1–19, 2015.
[25] X. He, T. Chen, M.-Y. Kan, and X. Chen, “Tirrank: Review-aware explainable recommendation by modeling aspects,” in Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015, pp. 1661–1670.
[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2018.