CMF-SMR: Convolutional matrix factorization for sequential movie recommendations

Zihui Shi[1,3] and Yaoning Ge[2]

[1]Aberdeen Institute of Data Science and Artificial Intelligence, South China Normal University, Foshan, 528225, China
[2]Nanjing Foreign Language School, Nanjing, 210000, China

[3]2662043905@qq.com, bhs6570@outlook.com

Abstract. Recommender system (RS) has become an essential component of e-commerce, social media, and other online platforms. Collaborative filtering (CF) is one of the most commonly used techniques in RS that relies on user-item interactions to generate recommendations. However, CF suffers from the cold-start problem, sparsity, and scalability issues. To address these challenges, this work propose a hybrid system called Convolutional Matrix Factorization for Sequential Movie Recommendations (CMF-SMR), which combines matrix factorization (MF) with convolutional neural networks (CNNs). CMF-SMR leverages the non-linear feature extraction capabilities of CNNs and the representation learning abilities of deep learning to enhance the accuracy and robustness of traditional MF-based RS. Specifically, CNNs and MF were used to respectively extract features from user-item interaction data and use them as input for learning user and item representations. The learned representations are then used to predict user-item ratings. This work evaluates the performance of our proposed method on two publicly available datasets, and the experimental results demonstrate that our method outperforms several state-of-the-art techniques in terms of accuracy, scalability, and robustness. Moreover, this work conduct evaluation metrics to demonstrate the accuracy of our proposed method. Overall, our proposed CMF-SMR provides a promising solution for addressing the limitations of traditional CF-based RS and can be applied in various domains, including e-commerce, social media, and personalized content recommendation systems.

Keywords: recommender system, convolutional neural networks, matrix factorization, collaborative filtering.

1. Introduction

Several online platforms, including e-commerce websites [1], online streaming services [2], social networks [3], and others, rely heavily on recommendation systems (RS). These systems aim to give a better user experience and make it simpler for users to identify relevant things by anticipating users' interests and providing customized recommendations. Collaborative filtering, one of the many recommendation system (RS) techniques, is a well-liked technique that predicts a user's preferences or ratings for an item based on the preferences or ratings of other users. Collaborative filtering's core tenet is that people who share similar preferences in the past are likely to share them in the future.

The program first determined how similar people were using a similarity metric, such as Pearson correlation or cosine similarity, before employing collaborative filtering. Selecting comparable people and things in RS also involved the use of the K-Nearest Neighbor method [4]. Later, a method called matrix factorization [5] was developed, which factors the user-item matrix into low-dimensional latent factor matrices and uses those matrices to capture the underlying preferences or ratings of users and things. However the issue of learning from complicated data and sequential data plagued each of these systems. Convolutional and recurrent neural networks have been found to increase RS performance by processing sequential data in order to address these issues. However, there are still some shortcomings, such as poor accuracy, poor robustness, and sparse data [6].

In this study, a novel hybrid recommender system is presented named CMF-SMR that integrates collaborative filtering with neural networks. The CMF-SMR combines CNN and MF and uses user and item interaction to build predictions for each movies. The final recommendation to users should therefore be the union of these two projected lists. Two MovieLens public datasets, ml-100k and ml-1m, were used for experiments. This work select MF and CNN as the reference points. According to the experiment's findings, this new model beats all baselines and is feasible when considering Mean Absolute Error (MAE), Mean Squared Error (MSE), and Normalized Discounted Cumulative Gain (NDCG).

2. Related work

2.1. Collaborative filtering for recommender system

User-based CF was one of the earliest and most popular approaches to CF. In 1992, the concept of collaborative filtering in the context of personalized recommendations was introduced [7], which proposed a user-based CF algorithm called "The Tapestry System" that used a similarity measure to find the most similar users and recommend items based on their ratings. To find the similarity between users and items, Pearson correlation, cosine similarity, and K-Nearest Neighbor (KNN) algorithm [4] are used. One of the first studies on CF assessed several CF algorithms based on how well they performed for users, including correlation coefficients, computations of vector-based similarity, and the statistical Bayesian approach, and found that a Bayesian network with decision trees at each node and correlation performed better [8]. Later, the item-based approach [9] became very popular in practice. Hybrid approaches that combine CF with other techniques, such as content-based filtering or demographic filtering [10] were later introduced. Though the CF is straightforward and has strong interpretability, it has poor accuracy when users of items have few features.

2.2. Convolutional neural networks for recommender system

Convolutional neural networks (CNNs) have been put into use in various fields of machine learning such as computer vision, natural language processing, and speech recognition. Recently, CNNs have also been used in RS to learn representations of users and items and make personalized recommendations. A hybrid approach that combines collaborative filtering and content-based filtering with a CNN [11], called Hybrid-CNN, uses a CNN to capture the features of both users and items data and then combines them with a weighted sum to generate a recommendation. More recent work proposed a model called CoNet [12] that uses a two-dimensional CNN to capture the user-item interaction. The model also has a gating mechanism to manage information flow across the network and a self-attention mechanism to learn user and item representations. Overall, CNNs have shown promising results in recommendation systems, and have become an active area of research. To enhance the performance of these systems, numerous issues including scalability and interpretability still need to be resolved.

2.3. Matrix factorization for recommender system

Matrix factorization (MF) is a popular approach in recommender systems, and there has been a significant amount of research in this area. A well-known matrix factorization method for recommendations is singular value decomposition (SVD) [13]. The user and item factors are represented by two low-rank matrices in this method, and the dot product of these factors yields the projected ratings. Non-negative matrix factorization (NMF) [14] is another matrix factorization technique that has been used for recommendation. This approach learns non-negative user and item factors, which can be interpreted as topics or features of the items. Probabilistic matrix factorization (PMF) [15] as a variant of matrix factorization, modeling the rating distribution using a probabilistic approach. This technique assumes that the ratings follow a Gaussian distribution and learns the user and item factors using maximum likelihood estimation. Overall, matrix factorization is a popular and effective approach for recommender systems. However, there are many challenges in developing a good matrix factorization model such as sparsity of data, scalability, and cold start problem.

3. Preliminaries

3.1. Definitions

3.1.1. User-item data. The sequential interaction information between users and items $(u, i) = ([u_a, i_a], [u_b, i_b], \dots, [u_z, i_z])$ and the rating $R = (r_1, r_2, \dots, r_n)$ is taken as input, where (u_x, i_x) represents a random user and the movie that the user has watched, r_n is corresponding to the rating of (u_x, i_x).

3.1.2. Movie recommendation system. In the context of recommendation systems model, the input is a sequence of user-item interactions. Each interaction is represented as a tuple (u, i), where u represents a user and i is an item. The model's goal is to foretell the item (u,i+1) that the user would probably engage with next based on their past interactions.

3.2. Problem statement

A particular kind of recommendation system called a "movie suggestion" suggests products to customers based on their prior activities or behaviors in an effort to foresee the movies they will most likely interact with in the future. Let $U = \{u_1, u_2, \dots, u_n\}$ be the set of all users, and $I = \{i_1, i_2, \dots, i_m\}$ be the set of all items. Let H be the embedding dimension for users and items. Then, each user u and item i can be represented as a d-dimensional vector: $e_u = \{e_{u_1}, e_{u_2}, \dots, e_{u_H}\}$ represents the embedding for user u. $e_i = \{e_{i_1}, e_{i_2}, \dots, e_{i_H}\}$ represents the embedding for item *i*. To construct the input sequence for the model, this work concatenate the embeddings for each user-item interaction:

$$X = \{e_{i_1}{}^{(u)}, e_{i_2}{}^{(u)}, \dots, e_{i_H}{}^{(u)}\}, \tag{1}$$

where $u \in U$ and $e_{i_n} \in e_i$. Passing this sequence through the model, an output sequence $Y = \{y_1, y_2, \dots, y_m\}$ can be obtained, representing the items the user u will probably interact with next.

4. Methodology

4.1. Matrix factorization

By using matrix factorization (MF), the original huge matrix is divided into the sum of two smaller matrices. R_{ij} is the rating of movie *j* given from user *i*. This work factorize the user-rating matric into the product of two matrices: $U_{n \times k}$ and $V_{k \times m}$, which can be illustrated by:

$$R_{ij} = U_{n \times k} * V_{k \times m}, \tag{2}$$

where U is the matrix of user features while V is the matrix of item features. Multiplying the feature vector of users with the feature matrix of movies, the ratings from users towards each movie can be obtained, which can be illustrated by:

$$(u_1, u_2, \ldots, u_k) * \begin{pmatrix} v_{11} & \cdots & v_{1m} \\ \vdots & \ddots & \vdots \\ v_{k1} & \cdots & v_{km} \end{pmatrix} = (r_{1,} r_2, \ldots, r_m), \tag{3}$$

Using gradient descent, the level of iteration can be determined and find the best solution. The gradient is determined by:

$$\frac{\partial L}{\partial U_i} = -2 \sum_{j=1}^{n} R_{ij} \cdot v_j + 2\lambda U_i, \tag{4}$$

And the iterative function is illustrated by:

$$U = U - \alpha * \frac{\partial L}{\partial U}, \tag{5}$$

$$V = V - \alpha * \frac{\partial L}{\partial V}, \tag{6}$$

where *U* and *V* are matrices of users and items that are wanted. α is the learning rate, which controls how much adjust *U* and *V* in each iteration. A sluggish convergence can be caused by a poor learning rate, whereas a fast convergence might become unstable or diverge. $\frac{\partial L}{\partial U}$ and $\frac{\partial L}{\partial V}$ are the gradients of the cost function with respect to *U* and *V*, which are calculated using **Formula 4**.

4.2. Convolutional neural network for movie recommendation

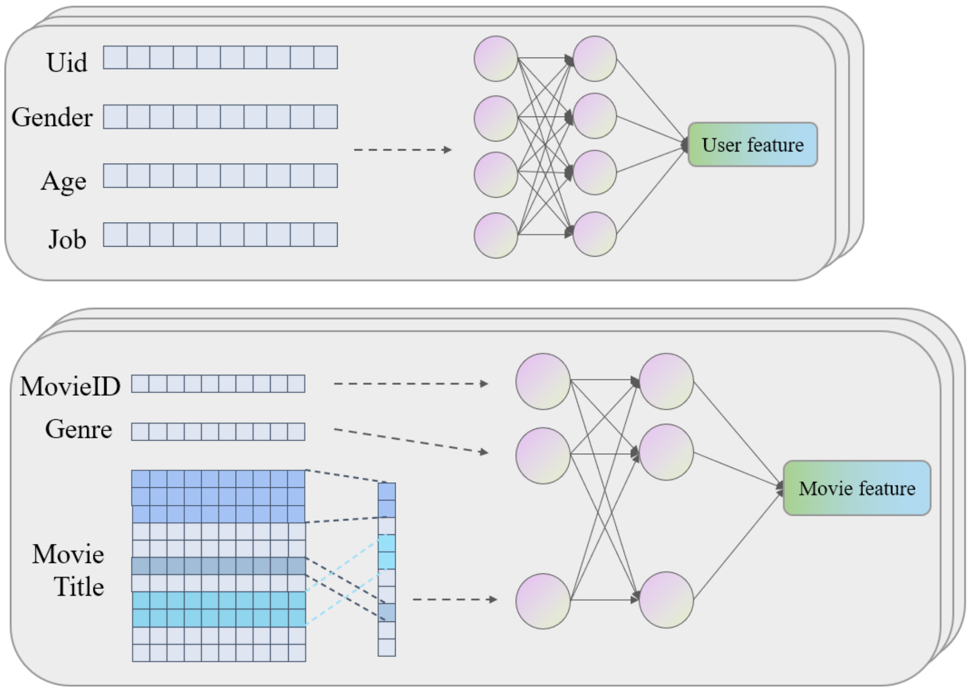

Figure 1. CNNs for movie recommendation (Owner-draw).

This work employ Convolutional Neural Networks (CNN) to capture user and movie feature (Figure 1). Convolutional, max-pooling, and fully connected layers are all included in CNN. Different from ordinary CNN models, this work adapt the CNN for NLP model to better fit the situation of movie recommendation. In the convolution layers, filters whose lengths are equal to the length of words and heights of 2, 3, and 4 are applied to slide over the input matrix, generating feature maps. Then max-pooling is performed to downsample the maps, creating feature vectors. This work also use average pooling to merge them into a fixed-length vector. The full connection layer is where the features gathered by the convolution and pooling layer are categorized. ReLU are used as the activation function to introduce non-linearity and avoid gradient vanishing problems, and mean squared error as the loss function to measure the difference between predicted ratings and true ratings, which is introduced in 4.4. Evaluation Metrics. By changing each neuron's weight in the network, the classification result can be obtained.

4.3. Optimization model: Hybrid collaborative filtering

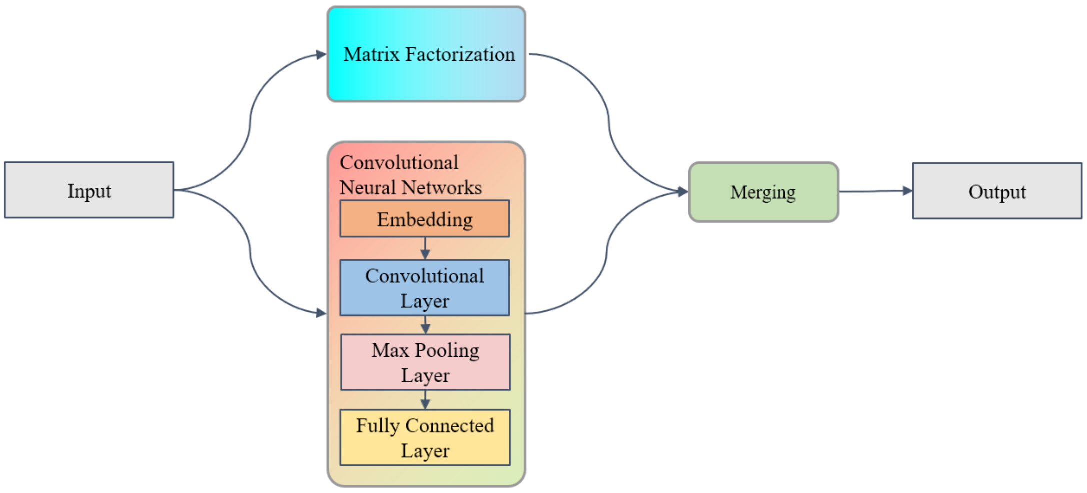

Figure 2. CMF-SMR framework (Owner-draw).

The overall goal is to predict the movie the user most possibly like to watch based on historical data. Hybrid collaborative filtering is used, as shown in Figure 2, to achieve this goal. Collaborative filtering (CF), based on the idea that people with similar historical record are likely to agree with each other’s choices in the future, utilizes the similarities between users and items to provide information. Hybrid collaborative filtering combines different methods of CF to provide a more accurate prediction. In the model, specifically, Convolutional Matrix Factorization for Sequential Movie Recommendations (CMF-SMR) is a model that combines matrix factorization(MF) and convolutional neural networks(CNN) to capture both the general and sequential preferences of users. Matrix factorization is used to learn the latent components of users and things from rating data, while convolutional neural networks are used to learn sequential patterns of users from movie sequences. It then combines there two sources of information to make rating predictions, which can be shown as:

$$p = \sqrt{P_1 P_2}, \tag{7}$$

Where P_1 is the prediction generated using MF model, P_2 is generated from CNN model, and p is the prediction of the CMF-SMR model. For movie recommendation, this work randomly select movies recommended by both models and keep the top ten movies.

5. Experiment

5.1. Datasets

Two MovieLens datasets are used to assess the hybrid model. One of the datasets that movie recommendation algorithms utilize the most frequently is MovieLens. As shown in Table 1, 100,000 ratings from 943 users have been given to 1,682 films in ml-100k, and 1000209 ratings from 6040 users have been given to 3900 movies. The dataset is available in different sizes, ranging from 100K to 20 million ratings. User information contains UserID, Gender, Age, and Occupation; movie information contains MovieID, Title and Genres; and rating information includes UserID, MovieID, rating, and timestamp.

Table 1. Datasets (Owner-draw).

Dataset	Users	Movies	Rating
ml-100k	943	1682	100000
ml-1m	6040	3900	1000209

5.2. Experimental settings
All the experiments were operated on NVIDIA Tesla K80 GPU running Ubuntu18.04, CUDA 11.2, cuDNN 8, NVCC, Tensorflow 2.5.0, Python 3.9 worked as a programming language and Jupyter Notebook worked as a programming environment.

5.3. Baselines
This work compares the proposed CMF-SMR to a set of commonly used baselines:

MF [13]: It is CF algorithm, which represents users, items, and their interaction in low dimensional matrix.

CNN [16]: Using feature matrices generated from convolutional neural network to provide predictions.

5.4. Evaluation metrics
To evaluate this new model and baselines, three of the evaluation metrics are employed. Normalized Discounted Cumulative Gain (NDCG@K), which is shown in equation (8)-(11):

$$NDCG = \frac{DCG}{IDCG}, \tag{8}$$

$$Gain = rel(i), \tag{9}$$

$$CG_k = \sum_{i=1}^{k} rel(i), \tag{10}$$

$$DCG_k = \sum_{i=1}^{k} \frac{rel(i)}{log_2(i+1)}, \tag{11}$$

where $rel(i)$ represent the correlation score of $item(i)$. IDCG (ideal DCG) is the best sort based on the descending rank of $rel(i)$.

Mean Absolute Error (MAE) is the mean of the difference between the predicted value and test value. MAE is calculated by:

$$MAE = \frac{1}{m}\sum_{i=1}^{m} |y_i - f(x_i)|, \tag{12}$$

where *m* is the size of the test sample, *y* is the real number, and *f(x)* is the predicted value.

Mean Squared Error (MSE) is the mean of the square of the difference between the predicted value and test value, which is calculated by:

$$MAE = \frac{1}{m}\sum_{i=1}^{m} (y_i - f(x_i))^2, \tag{13}$$

where *m* is the size of the test sample, *y* is the real number, and *f(x)* is the predicted value.

5.5. *Hyperparameter study*

5.5.1. *CNN model*

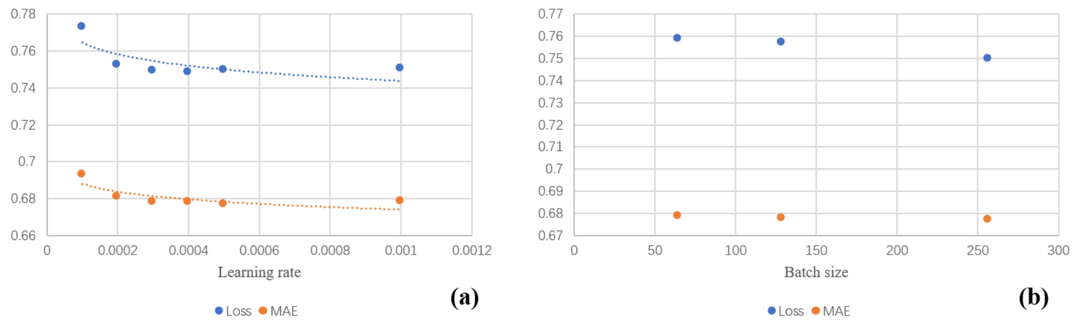

Figure 3. HyperParameter analysis on CNN model. (a) Learning rate. (b) Batch size (Owner-draw).

In Figure 3 (a), the x-coordinate shows the value of learning rate, while the y-coordinate illustrates the value of loss and MAE. The plot shows that both loss and MAE decrease rapidly when the learning rate increases from 0 to around 0.0005. This means that the model parameters are updated more efficiently and the error between actual and predicted values is reduced2. However, after reaching a minimum point around 0.0005, both loss and MAE start to increase slightly when the learning rate increases further. This means that the model parameters are updated too much and overshoot the optimum. Therefore, the learning rate of 0.0005 is chosen as the optimal choice.

In Figure 3 (b), with the different batch sizes of 64, 128, and 256, both the Loss and MAE do not change significantly. However, it seems that increasing the batch size from 64 to 256 leads to a lower loss (0.758501 to 0.749781) and a lower MAE (0.678708 to 0.676874), which means the model is more accurate and less biased when using a larger batch size. One possible reason for this is that a larger batch size allows it to learn from more diverse and representative samples of data in each iteration, which can improve its generalization ability.

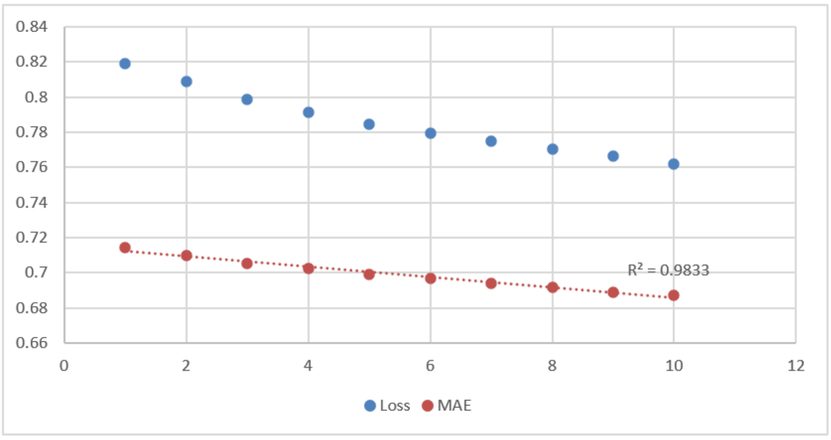

Figure 4. Epoch analysis on CNN (Owner-draw).

In Figure 4, The number of training epochs, with a batch size of 256 and a learning rate of 0.0005, is represented by the x-coordinate. The loss and MAE values are displayed in the y-coordinate. As the number of epochs increases, the loss and MAE keep dropping. The MAE value has a strong linear relationship with the number of epochs. It also seems that the improvement slows down after 10 epochs, which could indicate that the model is reaching a plateau or overfitting. Considering the efficiency and accuracy, this work takes epochs=5 in the model.

5.5.2. *MF model*

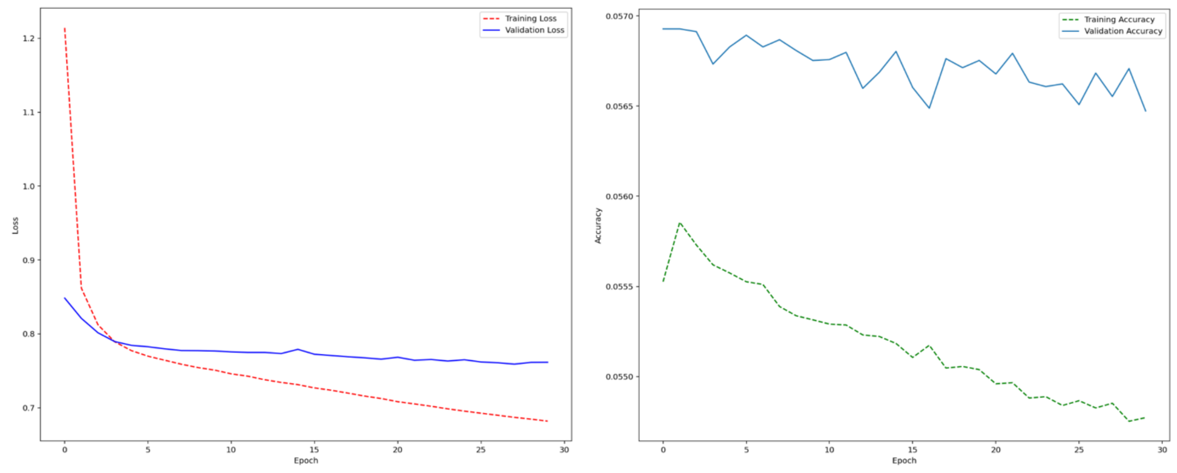

Figure 5. Epoch analysis on MF (Owner-draw).

The x-coordinate in Figure 5 denotes the number of epochs. The left figure's y-coordinate displays the loss value, while the right figure's y-coordinate displays the accuracy value. According to the data, the loss decreases as epochs increase, which means that the model is learning and improving its performance. However, it also seems that the improvement slows down after 10 epochs, which could indicate that the model is overfitting. By analyzing the loss and accuracy through each approach, a negative relationship between the times of iteration and the accuracy is discovered, indicating a problem of overfitting.

5.6. *Results analysis*

Table 2. Example of movie recommended by CMF-SMR for UserID=1 (Owner-draw).

MovieID	Title	Genre
567	Kika (1993)	Drama
162	Crumb (1994)	Documentary
1189	Thin Blue Line, The (1988)	Documentary
2393	Star Trek: Insurrection (1998)	Action|Sci-Fi
800	Lone Star (1996)	Drama|Mystery
1164	Two or Three Things I Know About Her (1966)	Drama
1243	Rosencrantz and Guildenstern Are Dead (1990)	Comedy|Drama
97	Hate (Haine, La) (1995)	Drama
1254	Treasure of the Sierra Madre, The (1948)	Adventure
3221	Draughtsman's Contract, The (1982)	Drama

As shown in Table 2, the ten movies are recommended by CMF-SMR model for user with the user id of 1. The genre of these movies matches the movies that user 1 rated, which shows that CMF-SMR can be applied to recommend movies for users.

Table 3. Comparison results (Owner-draw).

Model	**NDCG@10**	**MAE** (lower is better)	**MSE** (lower is better)
MF	0.884566351	0.680305	<u>0.7532299</u>
CNN	-	<u>0.676874</u>	0.8101309
CMF-SMR	-	0.657084	0.6993741

According to Table 3, CMF-SMR has the lowest MAE of 0.657084 and MSE of 0.6993741, which means it can predict the ratings more accurately than MF or CNN. This could be because CMF-SMR can capture both the general and sequential preferences of users, which are important factors for movie recommendations. By using convolutional neural networks, CMF-SMR can extract features from movie sequences that reflect the user’s changing interests and tastes over time. By using matrix factorization, CMF-SMR can also incorporate user-item interaction data that reflect the user’s overall preferences.

MF has a higher MAE of 0.680305 and MSE of 0.7532299 than CMF-SMR, which means it has a larger prediction error and lower accuracy. This could be because MF only uses user-item interaction data and does not consider any sequential information that could affect the user’s preferences. In order to represent each user and thing, MF makes the assumption that a vector of latent factors may be used to represent each user and item. However, this assumption may not hold for all users and items, especially when there is temporal dynamics or context-awareness involved.

CNN has an even higher MAE of 0.676874 and MSE of 0.8101309 than MF or CMF-SMR, which means it has the worst prediction performance among the three models. This could be because CNN only uses movie sequences as input and does not consider any user-item interaction data that could affect the rating predictions. CNN is a type of neural network that uses convolutional layers to extract features from images or text data. However, convolutional layers may not be sufficient for capturing all aspects of movie sequences, such as genre, theme, mood, or quality.

In summary, CMF-SMR is better at predicting ratings than MF or CNN because it uses both matrix factorization and convolutional neural networks to capture both general and sequential preferences of users. MF is worse than CMF-SMR but better than CNN because it only uses user-item interaction data without any sequential information. CNN is worse than both MF and CMF-SMR because it only uses movie sequences without any user-item interaction data.

6. Conclusion

In this work, a hybrid recommender system CMF-SMR using Matrix factorization (MF) and Convolutional neural networks (CNNs) approach was proposed to learn user-item interactions and sequential patterns in movie-watching histories and rating histories. This work evaluated this approach on a publicly available movie rating dataset MovieLens. In terms of Normalized Discounted Cumulative Gain, Mean Absolute Error, and Mean Squared Error, the trial results demonstrated that the model performs better than the other baselines. In conclusion, the proposed CMF-SMR demonstrates the effectiveness of combining matrix factorization and convolutional neural networks for sequential movie recommendations. The findings of this work imply that the suggested approach may be used for other sequential recommendation tasks across different domains, such as book or music suggestions.

This model does, however, also have some restrictions and difficulties that will need to be resolved in further studies. First, the model relies on the availability and quality of movie-watching histories and rating histories, which may be sparse or noisy in some cases. Second, the model may not scale well to large-scale datasets or complex scenarios, as it requires a lot of computational resources and parameters to train and optimize. Third, this model may not be easily interpretable or explainable, as it uses black-box neural networks to learn latent features and patterns. Therefore, we plan to explore more efficient and robust methods to handle data sparsity and noise, such as using attention mechanisms or transformer. We also plan to investigate more scalable and interpretable methods to learn sequential preferences, such as using recurrent neural networks(RNN). Moreover, we plan to conduct more experiments on different datasets and domains to evaluate the generalizability and applicability of this model.

References

[1] Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing, 7*(1), 76-80.

[2] Covington, P., Adams, J., & Sargin, E. (2016, September). Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems* (pp. 191-198).

[3] Wang, Z., Sun, L., Zhu, W., Yang, S., Li, H., & Wu, D. (2012). Joint social and content recommendation for user-generated videos in online social network. *IEEE Transactions on Multimedia, 15*(3), 698-709.

[4] Dumais, S. (1992). Enhancing performance in latent semantic indexing (LSI) retrieval.

[5] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 42*(8), 30-37.

[6] Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender systems: introduction and challenges. *Recommender systems handbook,* 1-34.

[7] Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM, 35*(12), 61-70.

[8] Breese, J. S., Heckerman, D., & Kadie, C. (2013). Empirical analysis of predictive algorithms for collaborative filtering. *arXiv preprint arXiv:1301.7363.*

[9] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295).

[10] Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., & Riedl, J. (1999). Combining collaborative filtering with personal agents for better recommendations. *Aaai/iaai, 439*(10.5555), 315149-315352.

[11] Salakhutdinov, R., Mnih, A., & Hinton, G. (2007, June). Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning* (pp. 791-798).

[12] Hu, G., Zhang, Y., & Yang, Q. (2018, October). Conet: Collaborative cross networks for cross-domain recommendation. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 667-676).

[13] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 42*(8), 30-37.

[14] Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature, 401*(6755), 788-791.

[15] Mnih, A., & Salakhutdinov, R. R. (2007). Probabilistic matrix factorization. *Advances in neural information processing systems, 20.*

[16] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017, April). Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web* (pp. 173-182).