Personalized Recommendation Algorithm Based On Deep Learning

Zhang Qidan
THE HONGKONG POLYTECHNIC UNIVERSITY
24126201g@connect.polyu.hk

Guo Siye
THE HONGKONG POLYTECHNIC UNIVERSITY
24112709g@connect.polyu.hk

Abstract

The recommendation system have evolved from collaborative filtering to deep learning, but they still face some challenges. Current research generally lacks multidimensional assessments of accuracy, diversity, novelty, and coverage. This study attempts to evaluate the quality of the four recommendation algorithms by systematically comparing the traditional method with the new hybrid model.

1 Introduction

Personalized recommendations are a key component of modern digital platforms, and understanding the performance trade-offs between different algorithmic approaches is critical for both research and practical implementation. This article uses the MovieLens dataset to make personalized recommendations for users based on movie characteristics and user characteristics. We focus on combining traditional collaborative filtering methods(1) with neural networks to discover potential nonlinear relationships and bring better recommendations to users, and compare them with traditional collaborative filtering methods to judge the quality of each model based on RMSE, precision, recall, and F1 score.

Link to our group's presentation video:https://youtu.be/tdWp8g70WU0

2 Dataset

2.1 Data Source and Description

The dataset used by our group is the public dataset of MovieLens(2) (ml-latest-small) from April 2018. Links to its data: https://files.grouplens.org/datasets/movielens/ml-latest-small.zip . The dataset contains a total of 4 CSV files, which are explained in Table 1.

Table 1: Description of the dataset content

File	content
ratings.csv	100,836 ratings with consistent user/movie IDs.
tags.csv	Contains 3683 labels. The label is user-defined short text .
movies.csv	The type list includes 18 types (such as Action, Horror) and No Type.
links.csv	Used for cross platform links (MovieLens, IMDb, TMDb).

The MovieLens dataset(ml latest small), which is widely used in the research of the recommendation system. The dataset contains 100836 scoring records of 9742 movies by 610 users, ranging from 0.5

to 5.0 stars, and metadata information such as movie title, type, release year, etc. This data set is of moderate scale, which can not only support comprehensive algorithm performance testing, but also reflect the sparsity problem commonly existing in real user rating data. Its rich data provides a basis for evaluating the diversity, coverage and other indicators of recommendation results.

2.2 Data Source and Description

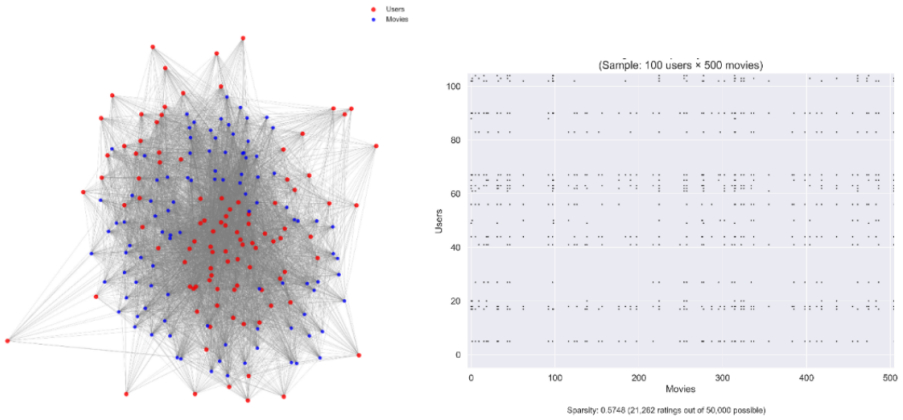

Figure 1: User-movie interaction graph

As shown in Figure 1, the user-item interaction graph reveals several important characteristics of the MovieLens dataset. First, the graph exhibits high sparsity, with most possible user-movie connections absent. Second, we observe heterogeneity in node degrees: some users (red nodes) have many connections, indicating highly active users, while others have few connections. Similarly, some movies (blue nodes) are highly popular with many ratings, while others have limited interaction. By randomly selecting 100 users and 500 movies, the matrix sparsity was 0.5748. However, the existence of high sparsity makes it difficult for new users-movies to obtain accurate recommendations due to insufficient interaction data, so we need to design a model to solve this kind of cold start problem.

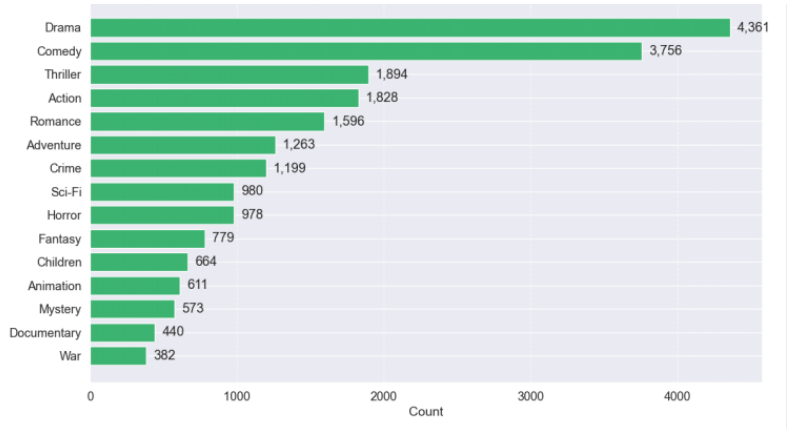

Figure 2: Top 15 Movie Genres

Figure 2 shows the overall distribution of movie genres in the dataset. Drama and comedy are the two main types of films, with 4361 and 3756 respectively. The distribution of genres in the data set is characterized by a long tail distribution. The first five genres (plot, comedy, thriller, action and love) account for most of the movies. This unbalanced distribution poses a challenge to the

recommendation system, which is mainly about how to ensure that small groups can get reasonable recommendation opportunities.

2.3 Data Preprocessing

We apply the following preprocessing steps:

1. First, load out the rating data and movie data, and filter out unpopular movies with a score of less than 20 by setting min_ratings=20 to avoid interfering with the model.

2. Create ID mappings to ensure the continuity of IDs and adapt them to the matrix factorization model(3).

3. Normalize ratings to [0,1] range for neural network models.

4. Split the dataset into 80% training and 20% testing sets by timestamp, stratified by user.

5. For graph-based models, create a bipartite user-item interaction graph.

3 Method

We formulate the recommendation task as a rating prediction problem. Given a set of users U and items I, we aim to predict the rating r_{ui} that user u would give to item i. We build 4 models: Matrix Factorization, Neural Collaborative Filtering(4), Long-Short Term Memory, and AttnGraphRec. The model was experimented to compare the accuracy of different models under the same dataset by adjusting the weight of model parameters, so as to select the optimal model.

3.1 Matrix Factorization

Matrix factorization is a traditional collaborative filtering algorithm, the principle of which is to decompose a matrix D into the product of U and V, that is, for a matrix D with a specific scale of m*n, the matrices U and V of scale m*k and n*k are estimated to approximate the values of matrix D as close as possible.

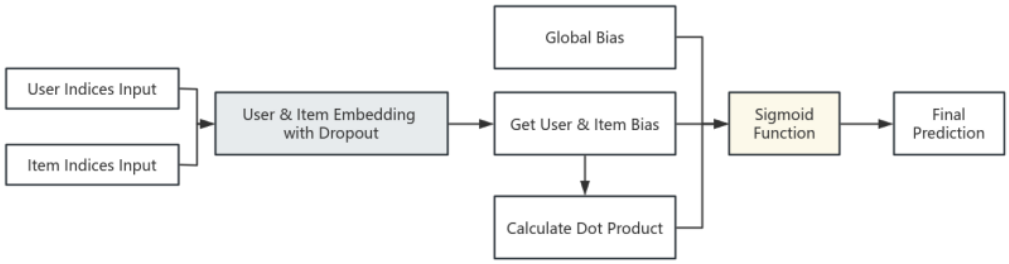

Figure 3: The Flowchart Of Training Matrix Factorization Model

As shown in Figure 3, the model obtains the corresponding embedding vectors from the respective embedding layers according to the indexes of users and movies, and applies Dropout to regularize representation to prevent overfitting, obtains bias items to capture personalized features, and calculates the dot product between user embedding and item embedding to measure the potential preference of users for items. Finally, the bias and dot product results are added to obtain the prediction results, which are mapped to 0-1 as the final output.

3.2 Neural Collaborative Filtering

Due to traditional collaborative filtering methods, the non-linear relationship between the user and the movie cannot be captured. Therefore, on the basis of calculating the dot product of the embedding vector, the neural network is used to introduce the Multi-Layer Perceptron (MLP)(5) as shown in

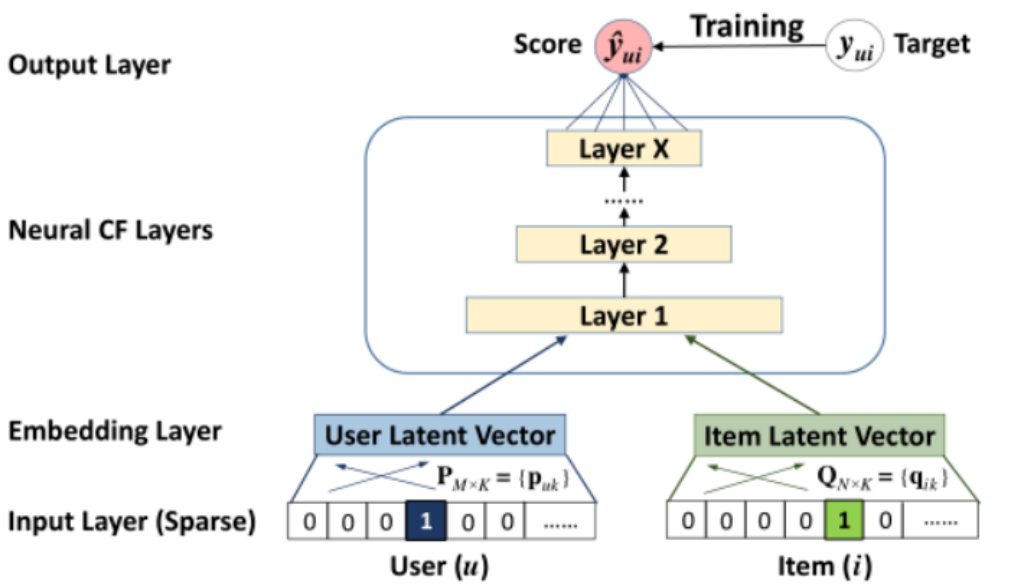

Figure 4: Neural Collaborative Filtering framework

Figure 4 to learn the nonlinear relationship between the user and the movie. The MLP model is a feedforward artificial neural network, which is composed of three layers of fully connected neurons (input layer, hidden layer, and output layer), and can learn the nonlinear feature representation of the input data by using functions such as ReLU and Sigmoid.

By introducing tags as an additional feature of the movie, for example, users give a high score to the movie "The Matrix", we can recommend movies of the same genre to users based on this movie tag. We set the number of neurons in each layer of MLP to 128, 64, and 32, respectively, and set the dimension of the embedding vector of user and movie to 64. By setting batch_size=256, we can obtain the parameter values for each layer in the model, as shown in Table 2. The model is

Table 2: Model Parameters

Parameter	Value
Embedding Layer	[256,64]
MLP First Layer	[256,128]
MLP Second Layer	[256,64]
MLP Third Layer	[256,32]
Output Layer	[256,1]

iteratively trained, and the weight of the model is adjusted by changing parameters such as the size of the batch_size and the learning rate, and the training is terminated when the loss in the validation set tends to be stable.The final evaluation index of the model is shown in Figure 5, which tends to stabilize when epoch=10.

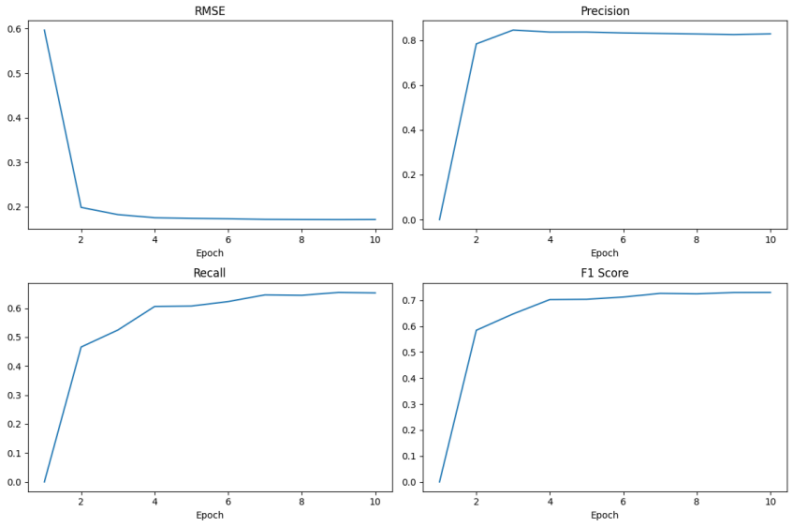

Figure 5: RMSE, Precision, Recall and F1 Score of NCF

3.3 LSTM-based Sequential Recommender

Sequential recommendation models capture temporal patterns in user behavior by modeling the sequence of user-item interactions. Our LSTM-based recommender leverages Long Short-Term Memory networks(6) to model sequential dependencies in user interaction history. The architecture of

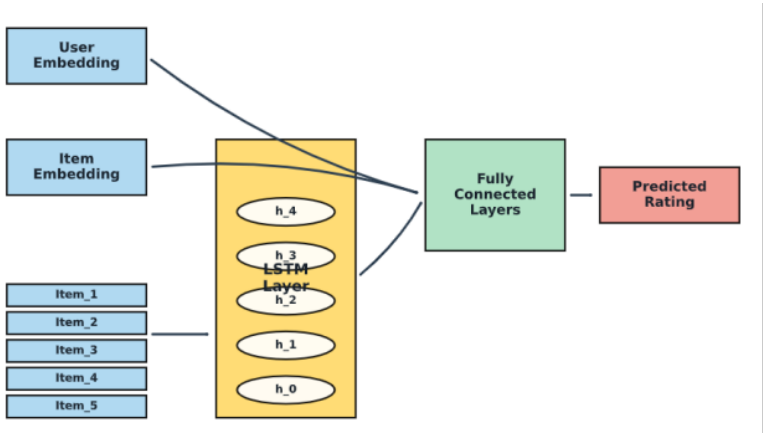

Figure 6: LSTM-based Sequential Recommender Architecture

the LSTM model in this study (Figure 6), includes several key components. User and item embedding layers with dimension 64. LSTM layer with hidden dimension 128. Prediction layers that combine LSTM output with the current item embedding. Dropout regularization with rate 0.2.

The LSTM layer processes the sequence of items a user has interacted with, learning temporal patterns in user preferences. For each prediction, we combine the LSTM's output with the embedding of the candidate item to predict the rating. This approach is particularly effective for capturing the evolution of user preferences over time and can potentially address the limitations of static models like Matrix Factorization and NCF.

3.4 AttnGraphRec

We propose AttnGraphRec, a hybrid recommendation model that combines attention mechanisms with graph neural networks(7). The model leverages the structural information in the user-item

interaction graph(Figure 7) while using attention to focus on the most relevant aspects of user preferences.

The AttnGraphRec model comprises several key components. First, the embedding layer maps users and items into a 64-dimensional space. Next, the graph neural network layer, utilizing graph convolutional networks (GCNs), captures structural information from the user-item interaction graph. The GCN layers facilitate information propagation between connected nodes, enabling the model to learn representations that incorporate neighborhood information. Then, the attention mechanism(Figure 8) dynamically weights different aspects of user and item representations based on their relevance to the current prediction, allowing the model to focus on the most pertinent features for each user-item pair. Finally, a multi-layer perceptron, which is a fully connected neural network, processes the combined representations from the GCN and attention layers to generate the final prediction. This bipartite graph structure(Figure 9) enables the model to capture collaborative signals

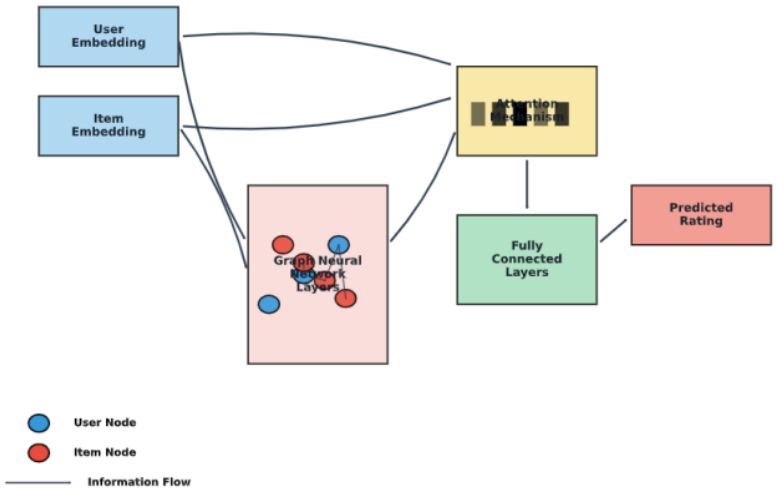

Figure 7: **AttnGraphRec Architecture**

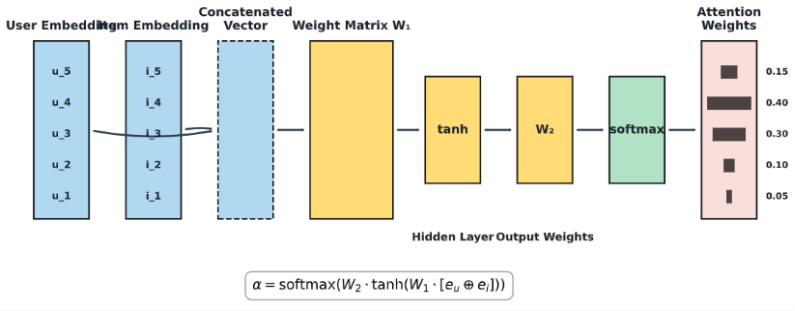

Figure 8: **Attention Mechanism in AttnGraphRec**

and structural patterns in the data. For example, if two users have rated similar items highly, their node representations will become similar after message passing through the GCN layers, leading to similar recommendations.

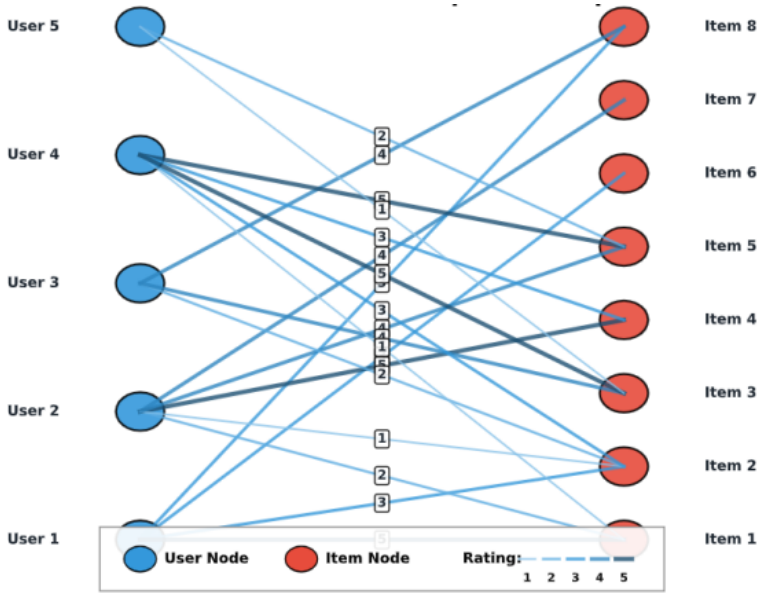

Figure 9: **Illustration of the bipartite user-item interaction graph**

4 Results and Analysis

4.1 Accuracy Comparison

Table 3 compares the performance of MF, NCF, LSTM and AttnGraphRec on four indicators.

Table 3: Result Comparison

Model	RMSE	Precision	Recall	F1 Score
MF	2.8674	0.8220	0.7402	0.7789
NCF	0.1711	0.8280	0.6579	0.7332
LSTM	2.9047	0.7280	0.6979	0.7126
AttnGraphRec	0.2087	0.6419	0.5650	0.6010

Through comparison, it is found that the NCF model is better than other models in scoring prediction accuracy, especially for the traditional MF model, indicating that its neural network structure can better capture the user-movie interaction. Although the prediction accuracy in the NCF model is high, there may be insufficient coverage that leads to poor recall effect, and the length of the recommendation list needs to be further adjusted, such as from top10 to top20. However, on the whole, the Precision, Recall and F1 scores of the MF model are at high values, indicating that MF is more suitable for sorting tasks than for regression problems. For the LSTM model, the time behavior data of the movie is not obvious in this model, the main reason may be that the timestamps are more closely distributed, and the LSTM model is more suitable for data with a longer time range. Due to the sparseness of the dataset, the number of node neighbors in the AttnGraphRec model is insufficient, and it is difficult for the attention mechanism to learn effective weights. Moreover, it only relies on the graph structure, and the lack of auxiliary information leads to poor training effect compared with other models.

4.2 Recommendation Result Analysis

Figure 10 shows the distribution of genres in the recommendation results generated by our model. Although drama plays a dominant role in the dataset (Figure 2), its proportion in the recommendation results is adjusted to 19.4%. This relatively balanced distribution shows that our model can effectively avoid the problem of over recommending mainstream genres and provide more diversified recommendations. In particular, the proportion of science fiction is significantly higher than its proportion

in the dataset, indicating that the model can improve the exposure of niche categories according to user preferences.

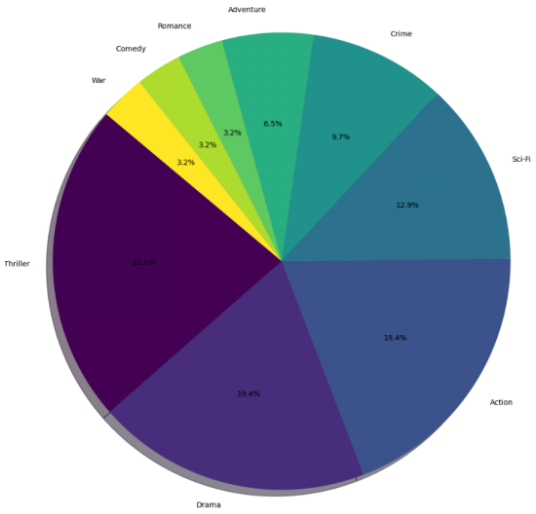

Figure 10: Genre Distribution in Recommendation

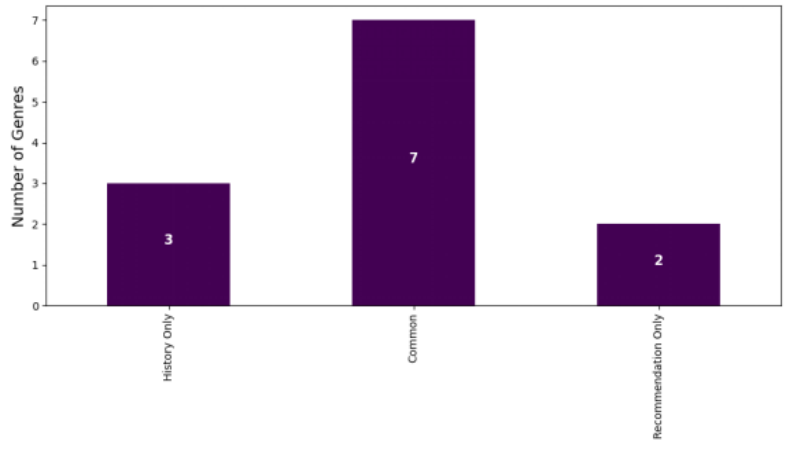

Figure 11: Genre Distribution between History and Recommendation

Figure 11 shows the distribution of genres. We find that 7 genres appear in user history and recommendation at the same time, while 3 genres only appear in user history and 2 genres only appear in recommendation. This distribution shows that our model not only maintains the continuity of user preferences (through common genres), but also provides moderate novelty (through genres only appearing in recommendations). This balance is essential to avoid the 'recommended echo chamber' phenomenon, that is, users are limited to the known preference range.

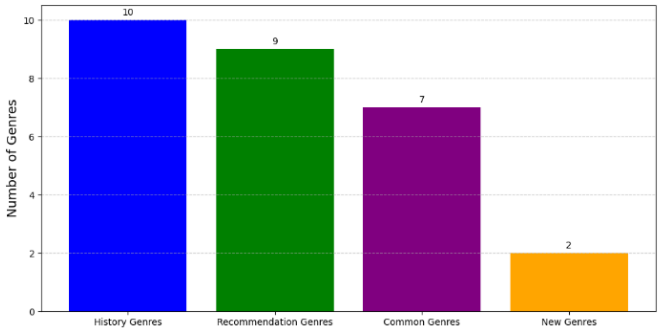

Figure 12: Genre Diversity Analysis

The performance of our model in terms of type diversity is shown in Figure 12. There are 10 types in the user history and 9 types in the recommended results. Of these, 7 types are common, indicating that the model retains the user's basic preferences. At the same time, the recommendation includes two new types, demonstrating the model's ability to introduce new content. This balance is essential to meet the user's known preferences and expand the user's range of interests. As illustrated in the

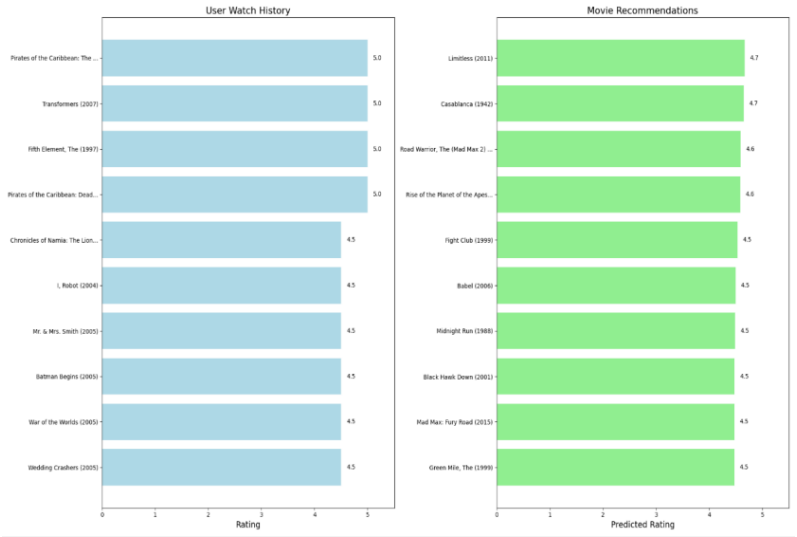

Figure 13: User recommendation results

Figure 13, the model is capable of generating personalized recommendations (right) based on movies that users have previously rated highly (left). It can be observed that the recommended results include both movies similar to those in users' historical preferences and new movies that may expand users' preferences.

References

[1] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, p. 5–53, Jan. 2004.

[2] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, Dec. 2015.

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

[4] W. Chen, F. Cai, H. Chen, and M. D. Rijke, "Joint neural collaborative filtering for recommender systems," *ACM Trans. Inf. Syst.*, vol. 37, Aug. 2019.

[5] A. Rana, A. Singh Rawat, A. Bijalwan, and H. Bahuguna, "Application of multi layer (perceptron) artificial neural network in the diagnosis system: A systematic review," in *2018 International Conference on Research in Intelligent and Computing in Engineering (RICE)*, pp. 1–6, 2018.

[6] R. Devooght and H. Bersini, "Long and short-term recommendations with recurrent neural networks," in *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, UMAP '17, (New York, NY, USA), p. 13–21, Association for Computing Machinery, 2017.

[7] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021.