Neural Collaborative Filtering (NCF) for Movie Recommendation on MovieLens 1M dataset

Jiaxuan Yu, Priyanjali Goel, Zixi Yang

1 Introduction

Recommender systems are vital for personalized content delivery, with collaborative filtering widely used. While Matrix Factorization (MF) captures latent user-item interactions, it is limited by linearity. Neural Collaborative Filtering (NCF) [2] overcomes this by combining the interactions through Generalized Matrix Factorization (GMF) and a Multi-Layer Perceptron (MLP) within a unified framework.

In this report, we implement Neural Collaborative Filtering (NCF) on the MovieLens 1M [1] dataset, covering the entire process from setting a rating threshold to model implementation and evaluation on the test set. We experiment with GMF, MLP, and their combined model, NeuMF. Our study focuses on analyzing the impact of different hyperparameter choices and model configurations. We evaluate performance using Recall@10 and NDCG@10 on the test set, providing insights into how various settings influence recommendation effectiveness.

2 Methods

2.1 Generalized Matrix Factorization (GMF)

The Generalized Matrix Factorization (GMF) model extends Matrix Factorization (MF) by learning latent vectors of users and items, $\mathbf{v}$ and $\mathbf{u}$, respectively. These embeddings transform high-dimensional, discrete user-item interactions into continuous, low-dimensional representations that capture underlying relationships. GMF then applies the Hadamard product between $\mathbf{v}$ and $\mathbf{u}$. The final prediction is obtained by applying an activation function a_{out} to the weighted sum of $\mathbf{m}$, as shown in the following equation:

$$\mathbf{m} = \mathbf{v} \odot \mathbf{u}, \quad \hat{y}_{ui} = a_{\text{out}}\left(h^T(\mathbf{m})\right)$$

where a_{out} is the activation function, and h denotes the edge weights of the output layer.

2.2 Multi-Layer Perceptron (MLP)

While NCF combines user and item features by concatenating them, hidden layers are then applied to the concatenated vector using MLP, allowing the model to capture non-linear interactions between user and item features, as defined by:

$$z_1 = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \end{bmatrix}, \quad z_L = a_L(W_L^T z_{L-1} + b_L), \quad \hat{y}_{ui} = \sigma(h^T z_L)$$

where z_1 is the concatenation of the user and item embeddings, for $L \geq 2$, W_L and b_L denote the weight matrix and bias of the L-th layer, $a_L(\cdot)$ represents the activation function of hidden layers, $\sigma(\cdot)$ is the output layer activation (usually $ReLu$), and h represents the edge weights of the output layer.

2.3 Neural Matrix Factorization (NeuMF)

Neural Matrix Factorization (NeuMF) combines GMF and MLP by learning separate embeddings for each. These embeddings are fused by concatenating the outputs of the final hidden layers, enabling the model to capture both linear and non-linear interactions for improved recommendations. Following the principles of GMF and MLP, NeuMF can be formulated as:

$$\hat{y}_G = \mathbf{v}^{\mathbf{G}} \odot \mathbf{u}^{\mathbf{G}}$$

[1] `http://grouplens.org/datasets/movielens/1m/`

$$\hat{y}_M = a_L(W_L^T a_{L-1}(W_{L-1}^T \cdots a_2(W_2^T a_1(W_1^T \begin{bmatrix} \mathbf{v^M} \\ \mathbf{u^M} \end{bmatrix} + b_1) + b_2) \cdots + b_{L-1}) + b_L)$$

$$\hat{y}_{ui} = \sigma(h^T \begin{bmatrix} \hat{y}_G \\ \hat{y}_M \end{bmatrix})$$

where $\mathbf{u^G}$, $\mathbf{v^G}$ are the user and item embeddings of GMF, $\mathbf{u^M}$, $\mathbf{v^M}$ are the embeddings in the MLP pathway, $\hat{y}_G$, $\hat{y}_M$ are the outputs of these two parts, and $\sigma(\cdot)$is the output layer activation of NeuMF.

3 Experiments

3.1 Dataset Splitting and Sampling Strategy

The dataset is divided based on interaction data: 70% for training, 15% for validation, 15% for testing. For each positive interaction in the training set, we select a certain number of negative samples from movies the user hasn't interacted with as a complement. Additionally, we sample 5 positive and 100 negative samples (which include both existing negative interactions and negative samples) per user from validation set to track the model's progress. The test set is constructed by including all movies that each user has not interacted with, in addition to their previously interacted data.

To prevent data leakage, negative samples from the training set are excluded from the validation and test sets, and those from the validation set are excluded from the test set.

3.2 Experimental Settings

To ensure a fair comparison, all experiments are conducted independently under consistent training conditions. Each model is trained for a maximum of 50 epochs, with early stopping based on validation performance using a patience of 20, and a batch size of 32.

NeuMF variation Due to the difference in labeling compared to the original paper, we adjusted the model to enhance optimization and stability. We use Kaiming initialization [1], drawing weights from a normal distribution with variance scaled by the number of input units to help mitigate vanishing or exploding gradient issues. Additionally, we implement a learning rate scheduler [3] to dynamically adjust the learning rate during training for better convergence.

Rate setting threshold As required, ratings above 4 are considered positive interactions. However, we believe this threshold may be too strict, limiting the model's capacity to learn the positive sample distribution. To evaluate the impact, we test three threshold settings:

- **Threshold = 4**: Ratings greater than 4 are labeled as positive, following the required setting.
- **Threshold = 3**: A more lenient threshold, where ratings greater than 3 are treated as positive.
- **Exclusion of Neutral Ratings (Rating = 3)**: Ratings of 3 are ignored from labels.

Dataset splitting strategy To evaluate the performance, we first compare two data split methods:

- **Random Split**: The dataset is randomly split using all interaction data per user with 70% for training, 15% for validation, 15% for testing.
- **Split by Positive and Negative Interactions**: The dataset is divided into positive and negative interactions, which are then split with ratios 70%, 15%, and 15% seperately.

Hyperparameter Ablation Experiment We conduct an ablation study to assess the impact of hyperparameters on model performance, testing various values to determine the optimal configuration. Since the last hidden layer acts as the predictive factor [2], we ensure consistency between the MLP and GMF components, fixing the MLP architecture based on the optimal hidden layer and negative sample count. For example, with a predictive factor of 8, the MLP architecture follows a halving pattern, i.e., [16, 8]. The experimental configuration is as follows:

- **Number of MLP Hidden Layers:** {1, 2, 3, 4}
- **Predictive Factor:** {8, 16, 32, 64}
- **Number of Negatives:** {1, 3, 5, 7, 10}

Model Comparison Pre-Training and Impact of K We compare the performance of the GMF, MLP, and NeuMF models using the optimal hyperparameters identified through the ablation experiment. Additionally, we explore the impact of pre-training and different values of K on NDCG@K and Recall@K.

3.3 Evaluation Metrics

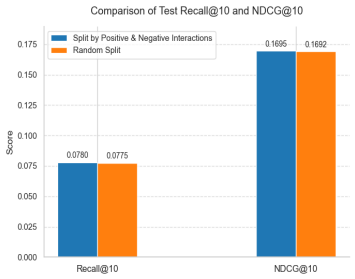
For the rate-setting threshold experiment, we use the F1 score to evaluate performance, as different thresholds affect the data distribution, making precision and recall insufficient for fair comparison.

For all other experiments, NDCG@10 and Recall@10 are computed using sampled validation data to track the model's progress. For the final evaluation, we sample all non-interacted movies per user in the test set as negative samples and compute Recall@10 and NDCG@10 to simulate real-world scenarios.

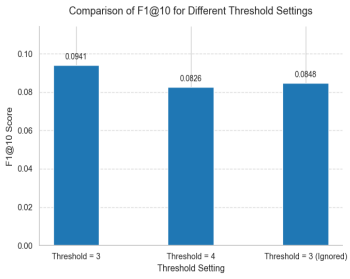
4 Experiment Results

4.1 Dataset Splitting Strategy

The results in Fig.1a show minimal differences in Recall@10 (0.0780 vs. 0.0775) and NDCG@10 (0.1695 vs. 0.1692), indicating that the data splitting strategy has little impact on model performance. Thus, we proceed with the random split for subsequent experiments.



(a) Split comparison



(b) Threshold comparison

Metric	T3	T4
Total interactions	1000209.0	1000209.0
Total users	6040.0	6040.0
Total items	3952.0	3952.0
Sparsity (All)	0.9581	0.9581
Sparsity (Positive)	**0.965**	**0.9759**
Total positive interactions	**836478.0**	**575281.0**
Total negative interactions	**163731.0**	**424928.0**
Removed users	**2.0**	**6.0**

(c) Dataset statistics

Fig. 1: Overall comparison of dataset statistics, thresholds, and split strategies. The two data splitting strategies yield comparable performance. Setting the threshold to 3 achieves the highest F1@10 (0.1033), while the third figure further shows dataset statistics with different thresholds.

4.2 Rate Setting Threshold

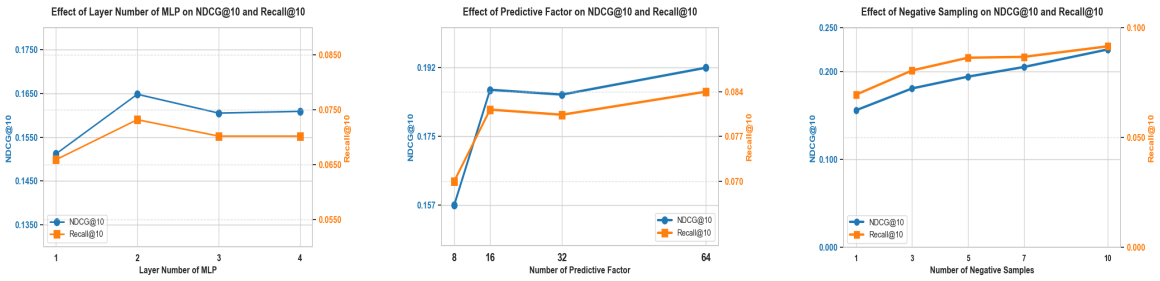
Threshold comparison As shown in Fig. 1b, Setting the threshold to 3 achieves the highest F1@10 score, suggesting that it is informative and positively contributes to model performance. In contrast, raising the threshold to 4 likely excludes some relevant positives. Similarly, ignoring class 3 removes potentially useful signals, reducing the model's ability to distinguish positives.

Dataset statistics Comparing the datasets split by thresholds 3 (T3) and 4 (T4) in Fig.1c reveals a notable shift in the distribution of interactions. T3 has 836,478 positive interactions, while T4 has only 575,281. Consequently, positive sparsity increases from 0.965 in T3 to 0.9759 in T4, indicating a much sparser distribution of positive feedback, leading to a less representative set for model training.

4.3 Hyperparameter ablation experiment with NeuMF

Number of MLP Hidden Layer Fig.2a illustrates that performance improves when increasing the number of layers from 1 to 2, with both metrics reaching their peak at 2 layers. However, as the layers increase beyond 2, both metrics decline. Although a shallow network may underfit, a deeper MLP introduces optimization difficulties or overfitting, especially as NeuMF tends to overfit when monitoring

performance on the validation set, resulting in degraded generalization. Therefore, a moderate depth (2 layers) strikes the best balance between model capacity and generalization on the test set.



(a) Layer number (b) Predictive factor number (c) Negative sample number

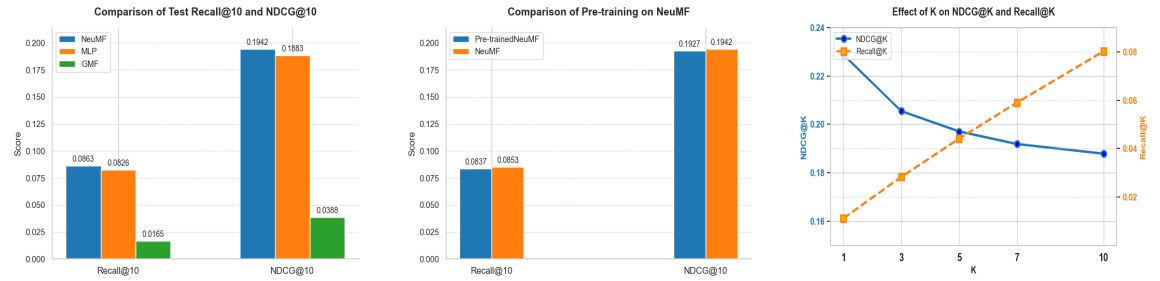
Fig. 2: Results of hyperparameter ablation experiment. A moderate number of MLP layers and negative samples yields better results, with a higher predictive factor consistently improving test set performance.

Number of Predictive Factors As Fig.2b demonstrates, with the number of predictive factors increasing from 8 to 64, both metrics show a clear improvement, particularly from 8 to 16. As the predictive factor is the last hidden layer, which can represent the model's capacity, a larger latent dimension enhances the model's capacity to learn user-item interactions. While performance improvements become marginal after 16 predictive factors, the configuration with 64 factors consistently achieves the best results. Therefore, increasing model capacity ultimately contributes to better overall performance.

Number of Negative Sample In Fig.2c, as the negative samples number increases from 1 to 7, the evaluation results show consistent improvement, reaching a peak at 10 negative samples. The increase in negative samples exposes the model to more diverse non-interactions, helping it better distinguish between positive and negative interactions. However, sampling 10 negative samples per positive sample is computationally expensive, and since performance does not improve significantly beyond 5 negative samples, we opt to use 5 negative samples for efficiency, as it yields comparable results with lower computational cost.

4.4 Model Comparison, Pre-training and Impact of K

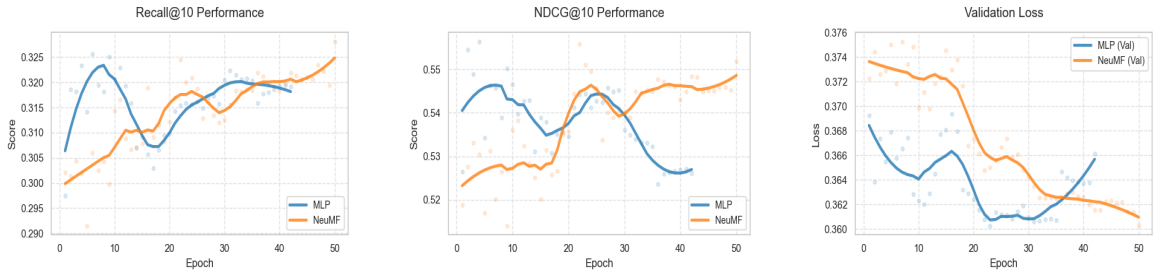
Effect of Pre-Training Fig. 3b illustrates that pre-trained NeuMF(trained using pretrained MLP and GMF) achieves marginally lower scores than standard NeuMF (trained from scratch) on both Recall@10 and NDCG@10, suggesting that while pre-training provides a warm start, it may also constrain the optimization process by limiting the flexibility of NeuMF to adapt its joint representation. The separately trained GMF and MLP components may converge to suboptimal solutions that are not well aligned when combined, leading to less effective interaction modeling.



(a) Model comparison (b) Effect of pre-training (c) Comparison with K

Fig. 3: (a) Comparison of Recall@10 and NDCG@10 across NeuMF, MLP, and GMF. (b) Pre-trained NeuMF vs. standard NeuMF. (c) Impact of varying predictive factor number K on test performance.

Impact of K Values on Recall@K and NDCG@K As shown in Fig.3c, as K increases, Recall@K improves from 0.0112 at K=1 to 0.0802 at K=10, indicating better coverage of relevant items. However, NDCG@K decreases from 0.2288 at K=1 to 0.1879 at K=10. This is because, although more relevant items are retrieved, their rankings become less optimal. Since NDCG is sensitive to the order of items, a decline in ranking leads to a decrease in its value, whereas Recall only measures whether relevant items are retrieved, irrespective of their ranking.



(a) Recall@10 over epochs (b) NDCG@10 over epochs (c) Validation loss trend

Fig. 4: Performance of NeuMF and MLP over training. NeuMF shows consistently strong improvements across all metrics, whereas MLP exhibits fluctuations and degrades in later stages. The solid lines show smoothed trends, dashed points mark original values.

Model Comparison Through hyperparameter ablation experiments, we determined the optimal configuration. NeuMF achieves the best performance on both Recall@10 (0.0826) and NDCG@10 (0.1917), slightly outperforming MLP, as shown in Fig. 3a. Given GMF's suboptimal performance, we further analyze the behavior of MLP and NeuMF on the validation set, as depicted in Fig. 4, to gain deeper insights into their strengths and limitations in different scenarios.

In Fig. 4a, NeuMF steadily improves its Recall@10 throughout the training epochs. In contrast, MLP fluctuates, initially declining before rising, but ultimately underperforms compared to NeuMF. As shown in Fig. 4b, NeuMF consistently improves in NDCG@10, while MLP experiences fluctuations and a continuous decline after around 25 epochs, which indicates a reduced ability to prioritize relevant items. Although both models suffer from overfitting, the validation loss in Fig. 4 indicates that NeuMF shows a more stable convergence and lower loss overall, which suggests that NeuMF generalizes better on the validation set.

Overall, NeuMF demonstrates superior performance across all key metrics because it combines the strengths of both linear and non-linear modeling. Unlike GMF, which only captures simple linear interactions, and MLP, which may struggle to model low-order patterns effectively, NeuMF jointly learns both types of interactions. This allows it to represent user-item relationships more comprehensively, leading to improved learning stability and stronger generalization throughout training.

5 Conclusion and Discussion

In this assignment we first revisit the rating threshold, with 3 offering the better trade-off. Building on this, we implement and evaluate NCF models on the MovieLens 1M dataset. NeuMF consistently shows superior performance across evaluation metrics. With carefully tuned hyperparameters—such as moderate MLP depth, 64 predictive factors, and five negative samples—NeuMF strikes a strong performance by capturing both linear and non-linear relationships between users and items.

Despite its advantages, NeuMF remains sensitive to architectural choices. Overfitting occurs with the model training process, limiting generalization. Although early stopping helps, it also hinders scalability on large datasets. Improving GMF and MLP through architectural enhancements and applying stronger regularization or adaptive training strategies remains a key direction for addressing these challenges in real-world recommendation systems.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification (2015), https://arxiv.org/abs/1502.01852
2. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering (2017), https://arxiv.org/abs/1708.05031
3. Wu, Y., Liu, L., Bae, J., Chow, K.H., Iyengar, A., Pu, C., Wei, W., Yu, L., Zhang, Q.: Demystifying learning rate policies for high accuracy training of deep neural networks (2019), https://arxiv.org/abs/1908.06477