A Preliminary Study on Enhancing Neural Collaborative Filtering with Self-Supervised Learning

210273B - L.C. Kariyawasam

October 4, 2025

Abstract

Neural Collaborative Filtering (NCF) is a foundational deep learning framework for recommendation. However, its performance can be limited by the sparse nature of user-item interaction data. This paper presents a preliminary study on enhancing NCF with an auxiliary self-supervised learning task (NCF-SSL). The goal is to improve the quality of user and item embeddings by leveraging a contrastive loss that enforces consistency between augmented views of the same entity. We conducted initial experiments on the MovieLens 1M dataset. The preliminary results are promising, indicating that NCF-SSL outperforms the standard NCF baseline on ranking-based metrics like Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). This suggests that self-supervised learning is a viable direction for regularizing collaborative filtering models and improving recommendation quality.

1 Introduction

Recommender systems are critical for personalizing user experiences on digital platforms. Collaborative Filtering (CF), which leverages past user-item interactions, remains a dominant approach. While traditional methods like Matrix Factorization (MF) were effective, they are limited by their use of a simple inner product to model interactions. He et al. [1] addressed this by proposing Neural Collaborative Filtering (NCF), a framework that uses a Multi-Layer Perceptron (MLP) to learn the user-item interaction function with greater non-linear expressiveness.

Despite its advantages, NCF's performance is often constrained by data sparsity. The supervision signal from observed interactions alone may be insufficient for learning robust representations that generalize well. This paper explores a method to enrich this learning process by introducing a self-supervised learning (SSL) task. We propose NCF-SSL, which augments the primary recommendation objective with an auxiliary contrastive loss. The core idea is that this auxiliary task provides an additional, powerful regularization signal, forcing the model to learn more meaningful and robust latent features for users and items.

Our primary objective is to conduct a preliminary empirical evaluation of the proposed NCF-SSL model against the standard NCF baseline to validate its potential.

2 Related Work

Since the introduction of Neural Collaborative Filtering [1], research in deep learning-based recommendation has rapidly evolved. Advancements can be broadly categorized into architectural innovations that enhance model expressiveness and new training strategies that improve learning from sparse data.

2.1 Architectural Advancements Beyond NCF

A primary focus of subsequent research has been to design architectures capable of capturing more complex dependencies than the standard MLP used in NCF.

Graph Neural Networks (GNNs)

Perhaps the most significant architectural shift has been the adoption of GNNs. This paradigm reframes the recommendation problem on the user-item interaction graph, allowing models to explicitly capture the

collaborative signal embedded in the graph's structure. Models like Neural Graph Collaborative Filtering (NGCF) [2] employ a message-passing mechanism where user and item embeddings are iteratively refined by aggregating information from their neighbors. This process enables the model to capture high-order connectivity, effectively encoding the signal that, for example, "users who interacted with items you liked also interacted with this other item." Subsequent work, such as LightGCN [3], demonstrated that simplifying the GNN architecture by removing non-linearities and feature transformations could lead to superior performance and efficiency, highlighting the centrality of the neighborhood aggregation process.

Other Deep Architectures

Researchers have also explored other neural architectures. For instance, ConvNCF [5] performs an outer product on user and item embeddings to create an "interaction map" and applies a Convolutional Neural Network (CNN) to learn local interaction patterns and high-order correlations from this map. Autoencoder-based models have also been used, where the network learns to reconstruct a user's interaction vector, with the compressed hidden layer serving as the user's latent representation.

2.2 Innovations in Training Objectives and Strategies

Parallel to architectural improvements, significant progress has been made in developing more effective training objectives and strategies to handle the challenges of implicit, sparse feedback.

Learning-to-Rank Objectives

The pointwise loss in NCF treats recommendation as a classification task for each user-item pair, which does not directly optimize for the ranking quality of a recommended list. To address this, many models adopt pairwise learning-to-rank objectives. The most prominent example is Bayesian Personalized Ranking (BPR) [6], which trains the model to rank an observed positive item higher than an unobserved (presumed negative) item. This approach better aligns the training objective with the ultimate goal of generating a ranked list of recommendations.

Self-Supervised Learning (SSL)

Recently, self-supervised learning has emerged as a powerful paradigm for representation learning, particularly in data-sparse domains like recommendations. The core idea is to create an auxiliary pretext task where the supervision signal is derived from the data itself. For recommendations, this is typically achieved through contrastive learning. The general framework involves: 1) creating two or more augmented "views" of the data (e.g., by applying dropout to embeddings or perturbing the user-item graph structure), and 2) training the model to maximize the agreement between different views of the same user/item while minimizing agreement with other users/items in the batch. Self-supervised Graph Learning (SGL) [7] successfully applied this concept to GNN-based recommenders, demonstrating significant gains in robustness and performance.

Our work is directly inspired by this trend. However, while SSL has proven highly effective for complex GNN architectures, its utility for simpler, foundational models like NCF remains less explored. This paper investigates whether the powerful regularization signal from a contrastive SSL task can directly benefit the original NCF architecture, providing a resource-efficient method to enhance its representation learning capabilities.

3 Proposed Method

Our proposed model, NCF-SSL, builds directly upon the NCF architecture. It maintains the dual pathways of Generalized Matrix Factorization (GMF) and a Multi-Layer Perceptron (MLP) but introduces a multi-task learning objective.

3.1 Data Augmentation via Dropout

To generate different "views" for the contrastive task, we use a simple stochastic augmentation method, embedding dropout. For each user u and item i, we create two augmented views of their embeddings

$(\mathbf{p}_u, \mathbf{q}_i)$ by applying dropout independently:

$$\mathbf{p}_u^{(k)} = \text{Dropout}(\mathbf{p}_u), \quad \mathbf{q}_i^{(k)} = \text{Dropout}(\mathbf{q}_i) \quad \text{for } k \in \{1, 2\}$$

where Dropout randomly zeroes out elements of the embedding vectors with a certain probability.

3.2 Multi-Task Objective Function

The model is trained by optimizing a composite loss function $\mathcal{L}$, which is a weighted sum of the recommendation loss $\mathcal{L}_{NCF}$ and the self-supervised contrastive loss $\mathcal{L}_{SSL}$.

$$\mathcal{L} = \mathcal{L}_{NCF} + \lambda \mathcal{L}_{SSL}$$

where λ is a hyperparameter balancing the two tasks.

Recommendation Loss ($\mathcal{L}_{NCF}$): This is the standard binary cross-entropy loss used in the original NCF paper, averaged over the two augmented views.

Contrastive Loss ($\mathcal{L}_{SSL}$): We use the InfoNCE loss to pull the two augmented views of an entity closer in the embedding space while pushing them away from other entities in the batch. For a user u in a batch $\mathcal{B}$, the loss is:

$$\mathcal{L}_{SSL}^{user} = \sum_{u \in \mathcal{B}} -\log \frac{\exp(\text{sim}(\mathbf{p}_u^{(1)}, \mathbf{p}_u^{(2)})/\tau)}{\sum_{v \in \mathcal{B}} \exp(\text{sim}(\mathbf{p}_u^{(1)}, \mathbf{p}_v^{(2)})/\tau)}$$

where $\text{sim}(\cdot, \cdot)$ is the cosine similarity and τ is a temperature hyperparameter. An analogous loss $\mathcal{L}_{SSL}^{item}$ is computed for items, and $\mathcal{L}_{SSL} = \mathcal{L}_{SSL}^{user} + \mathcal{L}_{SSL}^{item}$.

4 Experimental Setup

4.1 Dataset

The preliminary experiments were conducted on the widely used **MovieLens 1M** dataset. It contains 1,000,209 ratings from 6,040 users on 3,952 movies. Following common practice, we convert ratings into implicit feedback where any interaction implies a positive preference. As done by the authors of NCF, we randomly split the data into training and test sets, ensuring that each user has at least one interaction in the test set. All experiments were carried out without pretraining as the pretrained model weights were not made available.

4.2 Evaluation Metrics

We evaluate model performance using two standard top-K ranking metrics,

- **Hit Ratio (HR@10)** Measures the percentage of users for whom the correct held-out item is present in the top-10 recommendation list.
- **NDCG@10 (Normalized Discounted Cumulative Gain)** A measure of ranking quality that assigns higher scores for hits occurring earlier in the top-10 list. This metric accounts for the position of the correct item, rewarding models that rank it higher.

4.3 Implementation Details

We compare our proposed NCF-SSL against the original **NCF** [1] as our baseline. The code for the original NCF model provided by the authors but was re-implemented in PyTorch, as the provided code used outdated packages and versions. Both models were implemented using PyTorch and trained on Google Colab with NVIDIA Tesla T4 GPU.

4.3.1 MLP Architecture

The NCF paper describes the last layer of the MLP as the predictive factor, stating that it determines the model capability.

The Multi-Layer Perceptron (MLP) component follows a pyramid structure with three hidden layers such that the last layer has a dimension equal to the number of predictive factors (8, 16, or 32). Each of

the previous layers has double the number of neurons as the subsequent layer. The input layer receives the concatenated user and item embeddings. As done in the original implementation, an embedding size of double the predictive factor is used (i.e., 16, 32, or 64). For instance, if the predictive factor is set to 16, the MLP architecture will be $64 \rightarrow 32 \rightarrow 16$ neurons and the embedding size will be 32.

4.3.2 Training Configuration

We use a batch size of 256 as done by the original authors. However instead of using a fixed learning rate as mentioned by the authors, an exponential learning rate scheduler has been used with a decay rate of 0.85. This showed to improve convergence time over having a fixed learning rate. The models are trained for a maximum of 20 epochs with early stopping based on validation performance to prevent overfitting.

4.3.3 Hyperparameter Tuning

For NCF-SSL, we conducted systematic hyperparameter tuning experiments to determine optimal values for the self-supervised learning components:

- **Dropout rate for augmentation (p_{drop}):** We experimented with values in {0.1, 0.2, 0.4, 0.5} and found that 0.2 provided the best balance between creating meaningful augmented views while preserving essential interaction patterns. Given the small sizes of the embeddings, values below 0.1 did not introduce sufficient variability, while 0.5 led to excessive information loss.
- **Contrastive loss weight (λ):** We tested values in {0.2, 0.4, 0.6, 0.8} to balance the contribution of the contrastive loss with the primary recommendation loss. A weight of 0.8 yielded optimal performance, ensuring the contrastive objective enhances learning without overwhelming the primary task.
- **Temperature parameter (τ):** We explored values in {0.1, 0.5, 1} for the InfoNCE contrastive loss. A temperature of 0.5 was found to provide the right level of selectivity in the contrastive learning, making the model sufficiently discriminative between positive and negative pairs.

All hyperparameter experiments were conducted using cross-validation and the configuration achieving the highest average HR@10 was selected for final evaluation.

5 Preliminary Results and Discussion

The results of our preliminary experiments are presented in Table 1. The NCF-SSL model demonstrates a slight improvement over the standard NCF baseline across both evaluation metrics.
The values obtained for Hit Ratio (HR@10) and Normalized Discounted Cumulative Gain (NDCG@10) without SSL are inline with the values reported with the use of a 3 layer MLP (MLP-3) without pre-training, in the original NCF paper [1]. This validates our implementation of the baseline model.

Table 1: Performance of NCF and NCF-SSL with varying number of predictive factors on MovieLens 1M.

Factors	**NCF**		**NCF-SSL**	
	HR@10	**NDCG@10**	**HR@10**	**NDCG@10**
8	0.6824	0.4007	**0.6859**	**0.4060**
16	0.6972	0.4189	**0.7016**	**0.4210**
32	0.7004	**0.4251**	**0.7010**	0.4234

As shown, NCF with the self-supervised learning task (NCF-SSL) slightly outperforms the original NCF across all configurations of predictive factors (8, 16, and 32) except NDCG@10 for predictive factor of 32. However as the improvements are marginal, this alone cannot support the hypothesis that the auxiliary self-supervised task acts as an effective regularizer. While these results show some promise, they are preliminary and a more comprehensive analysis is required to fully validate the approach.

The effect of adjusting each of the hyperparameters specific to the self-supervised learning component were also studied in this preliminary phase. The dropout rate for augmentation (p_{drop}) and contrastive

loss weight (λ) were varied while keeping the other hyperparameters constant. The results for varying p_{drop} are shown in Figure 1 and for varying λ in Figure 2.

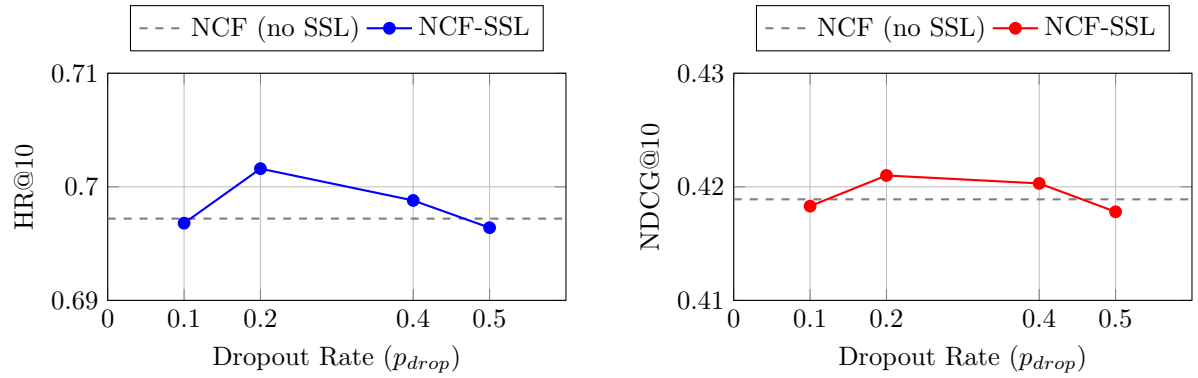

Figure 1: Effect of dropout rate (p_{drop}) on model performance. Dashed line indicates NCF baseline without SSL. Other hyperparameters kept constant, Predictive factors = 16, $\lambda = 0.8$, $\tau = 0.5$

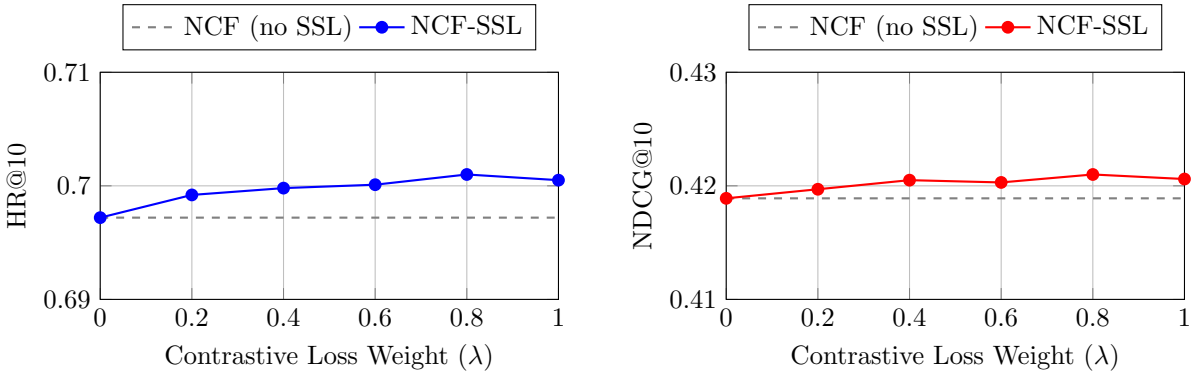

Figure 2: Effect of SSL weight λ on model performance. Dashed line indicates NCF baseline without SSL. Other hyperparameters kept constant, Predictive factors = 16, $p_{prop} = 0.2$, $\tau = 0.5$

The above figures indicate that both hyperparameters have an effect on model performance. A dropout rate of 0.2 appears to yield the best results, with performance declining at higher dropout rates. This suggests that while some level of augmentation is beneficial, excessive dropout may remove too much information, hindering the model's ability to learn effective representations. Similarly, increasing the contrastive loss weight (λ) up to 0.8 improves performance, but further increases lead to a slight decline. This indicates that while the self-supervised task is helpful, overemphasizing it can detract from the primary recommendation objective.

Overall, these preliminary results suggest that incorporating self-supervised learning into NCF has the potential to enhance recommendation performance. However, the improvements are modest and further investigation is needed. Future work will involve a more extensive hyperparameter search and experiments on larger and more diverse datasets including the Pintrest dataset, on which the original authors have benchmarked results, to fully assess the benefits of this approach.

References

[1] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. S. Chua, "Neural Collaborative Filtering," in *Proc. 26th Int. Conf. on World Wide Web (WWW)*, 2017, pp. 173–182.

[2] X. Wang, X. He, M. Wang, F. Feng, and T. S. Chua, "Neural Graph Collaborative Filtering," in *Proc. 42nd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, 2019, pp. 165–174.

[3] X. He, K. Deng, X. Wang, Y. Li, Y. Wang, and M. Wang, "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, 2020, pp. 639–648.

[4] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Attentional Collaborative Filtering: Recommending Images with Item- and Component-Level Attention," in *Proc. 40th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, 2017, pp. 953–956.

[5] X. He, C. He, M. Du, and R. C. K. Chan, "Outer Product-based Neural Collaborative Filtering," in *Proc. 11th ACM Int. Conf. on Web Search and Data Mining (WSDM)*, 2018, pp. 237–245.

[6] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian Personalized Ranking from Implicit Feedback," in *Proc. 25th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2009, pp. 452–461.

[7] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie, "Self-supervised Graph Learning for Recommendation," in *Proc. 44th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, 2021, pp. 726–735.