CPCS 583 Project Final Report: Recommendation System NGCF Enhanced by Random Sampling and Compound Risk Function

Helen Zheng
helen.zheng@yale.edu

Introduction

In the era of digital information, businesses face the challenge of navigating vast amounts of data to find best content recommendations for their customers. For example, MovieLens, being one of the most comprehensive platforms for movie reviews and information, encapsulates a wide range of transnational information (user-movie), user demographic data (user-user), and movie features (item-item). This project aims to: **build a robust model to accurately predict user preferences and provide top-notch recommendations** taking MovieLens data as the primary focus.

The mainstream recommendation models, such as Neural Graph Collaborative Filtering (NGCF), are subject to the following two issues, which I explored possible solutions to:

1. The recommendation systems are also constrained by data sparsity and cold start issue. In this project, I investigated: ***If augmenting training data by randomly sampling training datasets with replacement will reduce over-fitting and improve model performance on new sparse data?***
2. Most mainstream training methods involve back-propagation on one risk function. In this project, I explored: ***If back-propogating on sum of two risk functions will improve model performance?***

I applied the NGCF on MovieLens 100K predicting users' rating on movies, and on IMDB datasets predicting actors' likelihood in participating in movies, reveals the following results: 1. Random Sampling improved the training error but does not improve test model performance; 2. Compound Risk functions effectively improve model performance and generates a model that matches the baseline; 3. Deeper NGCF (4 layers) outperforms 3 and 2 layers.

Related Works

1. Neural Graph Collaborative Filtering NGCF [2]

The authors introduced NGCF as a graph neural network-based recommendation framework that captures the collaborative signal through embedding propagation, representing high-order connections within a user-item bipartite graph. Embedding defined as following (prediction formula given in **Proposed Approaches**):

$$e_u^{(k+1)} = \text{ReLU}\left(W_1 e_u^{(k)} + \sum_{i \in \mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}(u)|} \cdot \sqrt{|\mathcal{N}(i)|}} \left(e_u^{(k)} \odot W_1 + e_i^{(k)} \odot W_2\right)\right)$$

$$e_i^{(k+1)} = \text{ReLU}\left(W_1 e_i^{(k)} + \sum_{u \in \mathcal{N}(i)} \frac{1}{\sqrt{|\mathcal{N}(u)|} \cdot \sqrt{|\mathcal{N}(i)|}} \left(e_i^{(k)} \odot W_1 + e_u^{(k)} \odot W_2\right)\right)$$

- $e_u^{(k)}$ is the embedding of user u at the k-th layer.
- $e_i^{(k)}$ is the embedding of item i at the k-th layer.
- $\mathcal{N}(u)$ and $\mathcal{N}(i)$ represents the set of items that user u and user $i$has interacted with.
- W_1 and W_2 are weight matrices.

Advantages:

- **Exploiting High-Order Connectivity:** NGCF can exploit high-order connectivity in the user-item interaction graph, enabling it to uncover complex and indirect relationships between users and items.

- **Flexibility and Scalability:** NGCF can be adapted to various types of data and scales relatively well with the size of the datasets

Disadvantages:

- **Data Sparsity and Cold Start**: NGCF struggles with data sparsity and cold start problems (new users or items with limited interactions), a common challenge in recommendation systems.
- **Overfitting Risks:** NGCF can be prone to over-fitting, especially when the user-item graph is sparse or when there's a lack of sufficient training data.

2. Soft-Sampling [3]

This research done by Cui and his colleagues examines "soft sampling," an efficient approach for training large-scale deep neural networks with extensive datasets. Soft sampling involves randomly selecting a subset of data with replacement in each training epoch. The study provides a theoretical convergence guarantee for soft sampling on non-convex objectives, along with a convergence rate. The effectiveness of soft sampling is demonstrated through various machine learning tasks and network architectures, showing that it offers a better balance of accuracy and efficiency compared to coreset-based data selection methods.

Methodology

Basic Model Training

I started with basic model training: NGCF. The data from MovieLens is preprocessed to only contain edges that has rating bigger than or equal to 3. For NGCF, after Lth layer of embeddings, we concatenate all the embeddings:

$$e_u^* = e_u^{(0)}||...||e_u^{(L)};\ e_i^* = e_i^{(0)}||...||e_i^{(L)}$$

The **final prediction** is the inner product: $\hat{y} = {e_u^*}^T \cdot e_i^*$. For the MovieLens dataset prediction, each value in matrix shows the rating that the user will likely to rate the movie for each user-movie pair, on the scale from 1 to 5. For the IMDB dataset prediction, each value in matrix shows how likely an actor will participate in a movie, on scale from 0 to 1. For the basic model training step, I tested different number of layers (2, 3, 4) and find the most optimal value.

Improvement 1: Randomness by Sampling with Replacement

This idea is inspired by the image augmentation, such as cropping, rotating, and extra, usually performed before feeding images into CNN. It referenced to the study of Soft-Sampling in the **Related Works** section. I experimented with different proportion (60%, 70%, 80%) of random sampling with replacement to prevent the recommendation system from over-fitting to unseen datasets. This methods aims at solving the over-fitting, and cold start issues by applying soft-sampling to NGCF model. Such experiment has never been done in literatures I reviewed and has great impact if proven to be effective.

Improvement 2: Compound Risk Functions

In most of the codes I came across, researchers tend to use pairwise BPR loss to back-propagate NGCF: $L_{BPR} = -\sum_{(u,i,j)\in D} \log \sigma(\hat{y}_{ui} - \hat{y}_{uj})$, which faces couple shortages to the datasets I chose:

- **Negative Sampling:** BPR implicitly assumes that any item not interacted with is less preferred, which may not always be true. Negative Sampling can be obtained in datasets, such as MovieLens that involves different ratings, but does not apply to Actor-Movie interaction of IMDB datasets
- **Overfitting:** Due to the large number of parameters and the complexity of user-item interactions, models trained with BPR can be prone to overfitting

As the result, I decided to combine 2 error terms together to replace the L_{BPR} which can be applied flexibly to datasets without negative sampling possibility.

$$\textbf{Error Term 1:} MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\textbf{Error Term 2: } \text{Log Error} = \frac{1}{n}\sum_{i=1}^{n} log(y_i) - log(\hat{y}_i)$$

$$\textbf{Total Loss: } \text{Log Error} + MSE$$

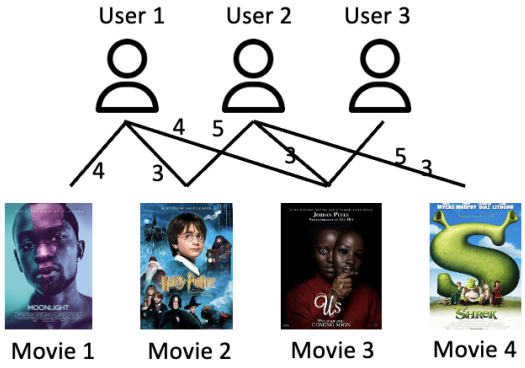

Figure 1: Illustration of graph structure MovieLens

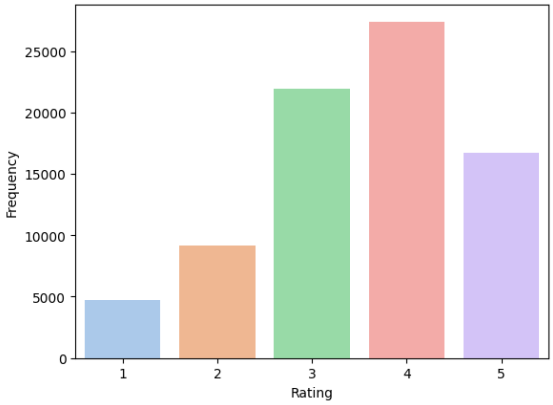

Figure 2: Frequncy by Ratings

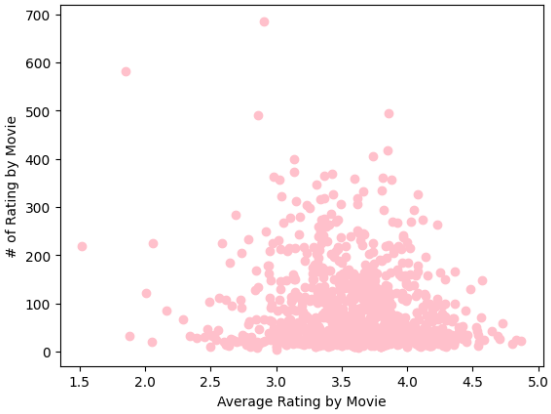

Figure 3: Average vs # of Rating by Movie

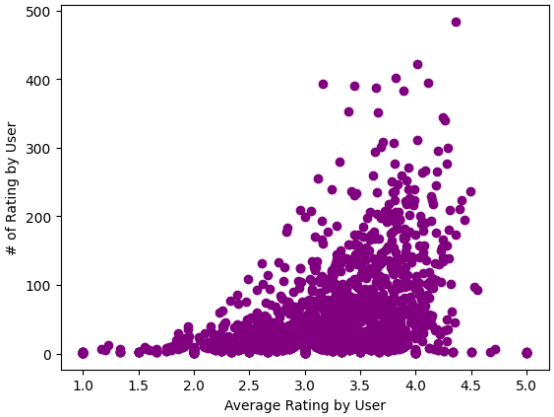

Figure 4: Average vs # of Rating by User

Experiment

I. Dataset

A. PyGeometric MovieLens100K Dataset (Heterogenous): LINK TO DATASET.

1. Feature and Nodes

This datasets have 2 types of nodes: **Movies** and **Users**, and 2 types of edges: **Directed** and **Undirected**. There are 16,000 edges in total: 8,000 directed edges and 8,000 undirected edges containing the extactly same features: Rating and Time. I will use undirected egdes for this project. Through data exploration, I decided to drop the time feature from edge features since the time span of the dataset is only around 1 year, which should not impact movie rating significantly. (See Figure 1 for data structure illustration.)

Table 1. Data Overview Movie Lens

Node/Edges	# of Nodes/Edges	# of Features
Overall	2,625/ 8000	N/A
Movie Nodes	1682	18
User Nodes	943	24
User-Movie	80000	1

2. Rating

Ratings are given from: 1, 2, 3, 4, 5. The average movie rating is 3.52835. As visualized in the Figure 2, the majority of ratings centered at 3 and 4, and rating 4 and 5 accounts for 55% of overall ratings. Therefore, I will use rating ≥ 3 as the threshold for user preference (rating ≥ 3 indicates user like this movie). Other information please see Figure 3 and Figure 4.

B. PyGeometric IMDB Dataset (Heterogenous): LINK TO DATASET.

Figure 5: Illustration of graph structure IMDB

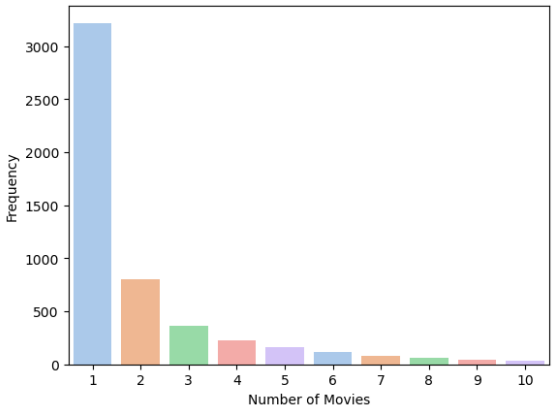

Figure 6: Actors' Movie Frequncy

1. Feature and Nodes

This datasets have 3 types of nodes: **Movies** and **Directors**, **Actors** and 2 types of edges: **Directed** and **Undirected**. There are egdes between actors & movies, and edges between directors & movies. Edges contains no features. I will use undirected egdes between actors and movies for this project, since there are very few directors per movies (See Figure 5 for data structure illustration.)

Table 2. Data Overview IMDB

Node/Edges	# of Nodes/Edges
Overall	11,616/ 17,106
Movie Nodes	4,278
Director Nodes	2,018
Actor Nodes	5,257
Actor-Movie	12,828

2. Actors

As illustrated in Figure 6, the movies that casted by actors is an exponential decreasing shape. Over 3000 actors participated in only 1 movie, around 800 participated in 2 movies and this frequency reduces to 2 digits for 10+ movies.

II. Training Details

Data Pre-processing for MovieLens

- Data Preparation: Extract the rating, user index, and movie index from the Movielens edges; Filter out ratings below 3 (considering only positive ratings).
- Splitting the Data to 20% Testing, and 80% Training
- Define a custom MovieLensDataset class inheriting from Dataset.
- Create 2 DataLoaders: one without sampler, the other with sampler = RandomSampler(replacement = True, num_samples = 80% training size)

Data Pre-processing for IMDB

- Data Preparation: Extract the Actor-Movie Edges from database; Manually create a label of 1 for actor-movies that exist an egde.
- The other steps are similar as above

Hyperparameters. (Table 3)

Table 3. Hyperparameters Testing

Hyper-parameters	Values Tested	Optimized Value
Embedding dimension	32/ 64/ 128	64
Optimizer	SGD/ Adam	Adam
Learning Rate	0.001/ 0.0001/ 0.01/ 0.005	0.001
Batch Size	32/ 64/ 124	124

III. Testing Results

Table 4. Testing Results (MSE) on 10 Runs Testing

Datasets	# of Layers	Training Data	Risk Function	Test Results
MovieLens	2	Non-Random	MSE Only	1.644 ±0.117
MovieLens	2	Non-Random	Total Loss	1.319 ±0.119
MovieLens	2	Random	Total Loss	5.494 ±0.116
MovieLens	3	Non-Random	Total Loss	1.104 ±0.035
MovieLens	4	Non-Random	Total Loss	0.825 ±0.024
IMDB	2	Non-Random	MSE Only	0.0386 ±0.0009
IMDB	2	Non-Random	Total Loss	0.0337 ±0.0005
IMDB	2	Random	Total Loss	0.0513 ±0.0022
IMDB	3	Non-Random	Total Loss	0.0386 ±0.0009
IMDB	4	Non-Random	Total Loss	0.0241 ±0.0006

I obtained the above (Table 4.) testing results with different combinations of datasets, number of NGCF layers, the randomness of training data, and risk functions for back-propagation. Most recent baselines for comparisons used BCF pair-wise error, which will not be suitable in this case. The below baselines are in RMSE for the MovieLens dataset, while baselines for IMDB datasets are unavailable online. Therefore, I will only discuss the testing results for MovieLens comparing to baseline and convert the RMSE to MSE.

Table 5. Baseline Errors (RMSE converted MSE) for MovieLens [1]

Models	Year	MSE Error
RSVD	2006	0.84
BPMF	2008	0.82
GSMF	2014	0.81

Through analyzing the testing results and comparing the testing results with the baseline error rates, we can derive the following insights:

1. **Compound Risk Function Benefits:** The models trained with a compound risk function, referred to as "Total Loss" in the table, generally show improved performance over those trained with just "MSE Only" across both the MovieLens and IMDB datasets.
2. **Layer Depth Impact:** Increasing the number of layers from 2 to 4 seems to have a positive effect on model performance. The test results show a trend of decreasing error as the number of layers increases. This could indicate that a deeper model is capable of learning more complex patterns in the data.
3. **Effect of Data Sampling Method:** Models trained with non-random (presumably structured or sequential) training data consistently outperform those trained with random sampling. This suggests that random sampling with replacement, often used to introduce stochasticity and prevent overfitting, does not necessarily lead to better model performance on our selected databases.
4. **Compared with Baselines:** When we look at the our best results: 0.825 for MovieLens data with 4 layers of NGCF, non-random training data, and total loss risk function, it matches the baselines given in Table 5, ranging from 0.81 to 0.84. Our training is limited to 4 layers given limited computing resources and time. By increasing the neural network depths, we could potentially beats the baseline provided, which implies our improved version of NGCF outperforms the baseline for recommendation systems on MovieLens data.

IV. Training Error Curve

Figure 7 and Figure 8 visualizes the training error (MSE and Log Error) by epochs for all 10 models illustrated in the testing result table. Order from left to right, from top to bottom matches the order in results table; Red line represents the MSE error, Blue Line represents the Log Error defined in **Methodology** Section. Through analyzing the training error, we derived the following insights:

1. **Initial Learning Phase:** For both datasets, there is a sharp decrease in error at the beginning of training across all graphs, indicating that the models learn quickly in the initial epochs.
2. **Stabilization and Overfitting:** In both sets of graphs, the curves generally flatten out as training progresses, indicating a stabilization in learning. Both Errors also tends to move together no matter backpropogate on the Total Loss (Red + Blue) or only the MSE (Red).
3. **Convergence Behavior**: For the MovieLens dataset, the curves are smoother, while for IMDB, the red and blue error curves for more model with 3 and 4 layers initially move in opposite direction, indicating some hyperparameters could be better optimized for these 2 models specificially.

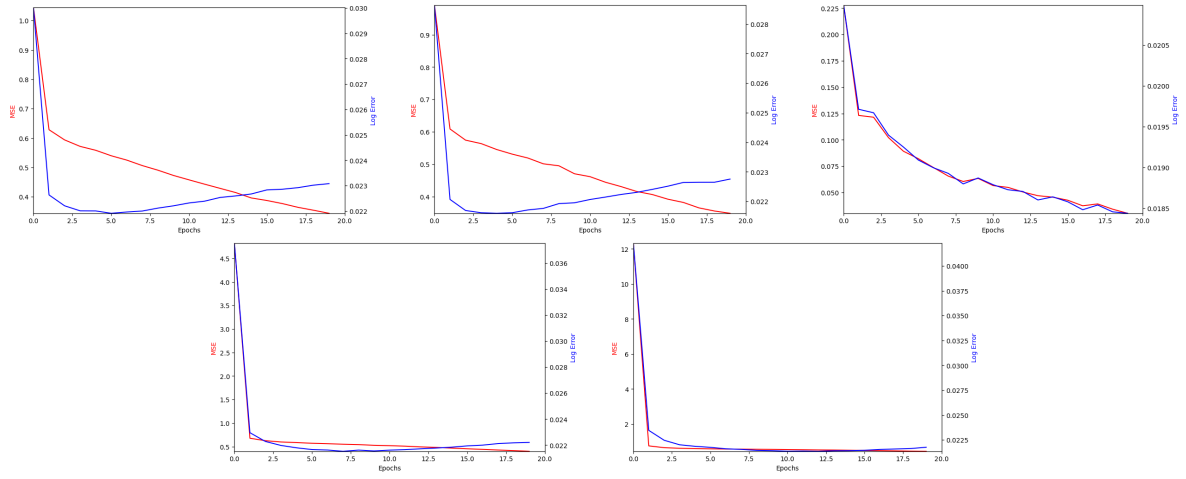

Figure 7: Training Error Curve by Epochs for MovieLens Data

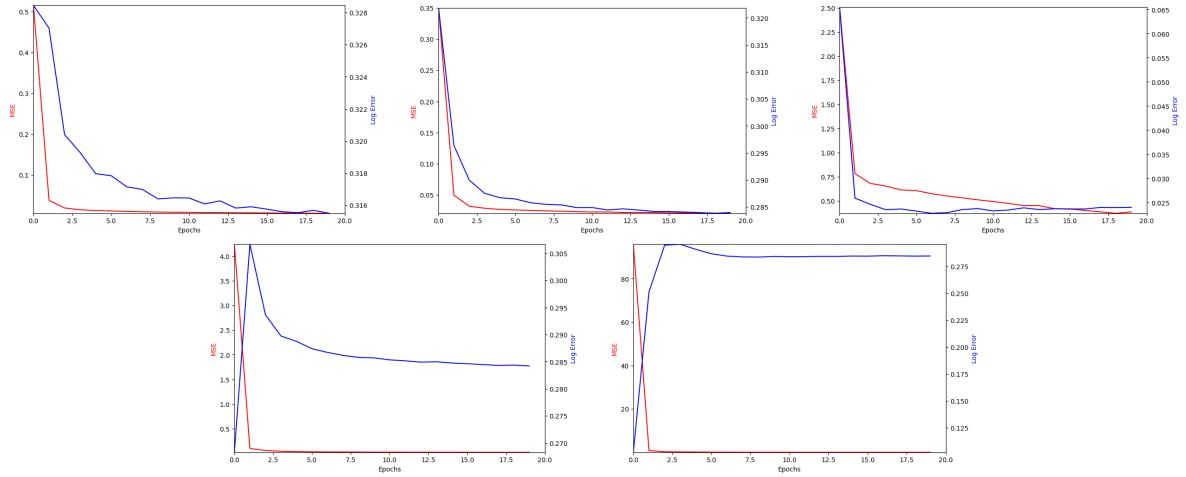

Figure 8: Training Error Curve by Epochs for IMDB Data

Conclusion (Github Link)

This project aims to improve recommendation systems, NGCF, by addressing the issues of data sparsity and the cold start problem. The project also experiments the use of compond risk functions instead of an individual risk functions.

The result demonstrates that incorporating a compound risk function, referred to as "Total Loss," into the training of recommendation models significantly enhances their performance on both the MovieLens and IMDB datasets when compared to traditional mean squared error (MSE) optimization. Additionally, the research indicates a positive correlation between the number of layers in a model and its performance, with a notable decrease in error rates as the layer depth increases, suggesting that more complex data patterns are better captured by deeper models.

The examination of different data sampling methodologies reveals that non-random data selection outperforms random sampling, challenging the common practice of using random sampling to avoid overfitting. Moreover, the optimized NGCF models achieve results that are on par with established baselines, with the potential to surpass them if the neural network's depth is increased beyond the current four-layer structure, given adequate computational resources.

Possible Next Step Exploration

1. Experiment deeper NGCF with 5 and more layers
2. Experiment different sampling techniques for random sampler, such as stratified sampling or importance sampling
3. Apply negative sampling on Datasets that does not naturally have negative samples

References

[1] Yehuda Koren Steffen Rendle, Li Zhang. On the difficulty of evaluating baselines. *Google Research*, 2019.

[2] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. *SIGIR'19: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019.

[3] Songtao Lu Wei Zhang George Saon Brian Kingsbury Xiaodong Cui, Ashish Mittal. Soft sampling for efficient training of deep neural networks on massive data. *ICLR 2023 Conference*, 2023.

Code Reference

1. https://medium.com/@meuleman.mathias/reproducing-neural-graph-collaborative-filtering-a8982c7d3df
2. https://github.com/xiangwang1223/neural_graph_collaborative_filtering/blob/master/NGCF/NGCF.py
3. https://github.com/huangtinglin/NGCF-PyTorch/blob/master/NGCF/NGCF.py
4. https://medium.com/@pytorch_geometric/link-prediction-on-heterogeneous-graphs-with-pyg-6d5c29677
5. https://medium.com/stanford-cs224w/recommender-systems-with-gnns-in-pyg-d8301178e377