Practical ML and DL
Assignment 2
Movie Recommender System

Roman Makarov
B-20-AI-01
o.makarov@innopolis.university

Introduction

The following is a report on my solution for Assignment 2 “Movie Recommender System” on Practical ML and DL course. It is organized into multiple sections, each dedicated to explaining a certain part of my solution.

Data analysis

During the data analysis stage, I examined the main data files and consolidated them into a single dataframe. Through plotting various graphs, I noticed the diversity in movie data, but a majority of the movies received ratings from fewer than 100 people. Subsequently, I decided to develop a model solely based on the provided ratings, without creating additional embeddings, given the potential scarcity of data for embedding creation. After visualizing the rating distribution, I observed a significant number of ratings at 3 or higher. Therefore, I chose to use this threshold as a basis for predictions, as attempting predictions for ratings of 4 or higher resulted in poorer model performance, perhaps due to a lack of data.

Model Implementation

As a backbone for my model, I chose a LightGCN with the latent dimension equal to 256. I have chosen this model as it stands among the SOTA models for recommendation systems and I wanted to understand its working principles deeper. After a number of tries, I came up with a final architecture that includes a total of 4 LGCN layers. I trained the model for 50 epochs, as further experimentation revealed that around this point, the model ceases to significantly improve.

Model Advantages and Disadvantages

The main advantages of the model are:

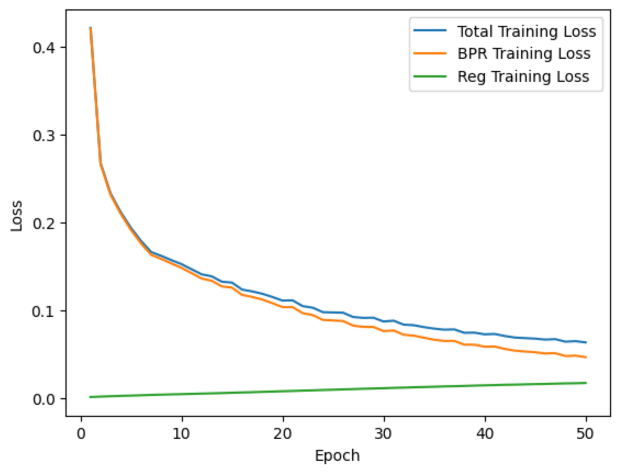
- Lightweightness
  LightGCN was created as a simplified version of GCN, so it can easily be fit into the memory.
- Fast training
  The training process with the parameters I specified earlier only takes 3.5 minutes.

The main disadvantage of the model is:

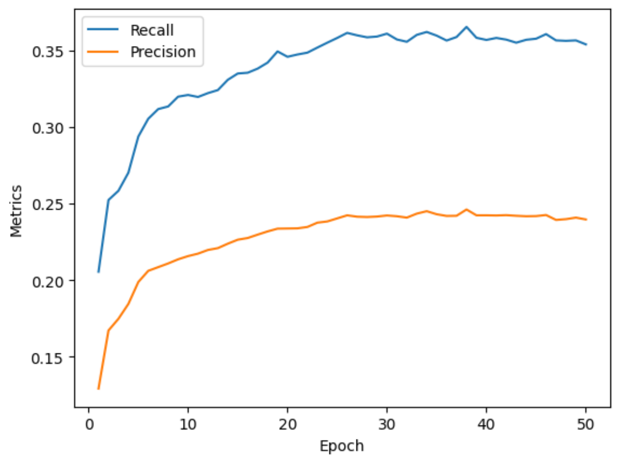
- Lack of Personalization
  The model is not fed with the information about the user or item, thus there is a chance that it could have performed better.

Training Process

I trained the resulting model for 50 epochs, utilizing a batch size of 1024 and a learning rate set to 5e-3. Throughout the training process, I measured the changes in BPR and Regularization loss. The evolution of these losses during training is illustrated below.



Furthermore, I assessed the recall and precision of the model. The graph depicting the trends of these two metrics is presented below.



Evaluation

To evaluate my model, I have found two other metrics:

- NDCG@K - a metric used to evaluate the effectiveness of a ranking algorithm by measuring the quality of the ranked results in comparison to an idealized ranking, and providing a normalized score that ranges from 0 to 1, with higher values indicating better performance.
- MAP@K - a metric used to assess the performance of a ranking algorithm by considering the precision of relevant items in the ranked list and averaging them. The resulting value is from 0 to 1, with higher values indicating better performance.

The model was evaluated on testing data that it has not observed during the training phase. The following section presents the outcomes of these measurements.

Results

The resulting model demonstrated the following results on a testing data:

● Recall@20:	0.35
● Precision@20:	0.24
● NDCG@20:	0.38
● MAP@20:	0.17

The results demonstrate a good performance of a recommender model compared to older methods that utilize KNN or SVC for predictions.