Individual Project Report

1. Introduction

In this report, I'll introduce the models and results about how am I aiming to develop recommendation models to recommend items for users.

2. Methodologies and models

2.1 Bayesian Personalized Pair (BPR)

Bayesian Personalized Ranking (BPR) is a machine learning algorithm primarily used for building recommendation systems. It is designed to optimize the ranking of items for individual users by leveraging implicit feedback (e.g., clicks, views, purchases) rather than explicit ratings.

Implicit Feedback: BPR utilizes **implicit** feedback, meaning it focuses on observed user behavior rather than relying on explicit ratings. But in this project, the dataset I got is **explicit** feedback, that's why the result of BPR is not so good I think.

Pairwise Ranking: The core idea of BPR is to train a model to rank a user's preferred items higher than the non-preferred ones. It does this by creating pairs of items where one is known to be interacted with by the user and the other is not.

Optimization: The training process involves optimizing a pairwise ranking loss function. The goal is to maximize the probability that the preferred item is ranked higher than the non-preferred item using a logistic function.

2.2 Weighted Regulation Matrix Factorization (WRMF)

Weighted Matrix Factorization (WRMF) is a collaborative filtering technique used for recommendation systems. It extends traditional matrix factorization by incorporating weights into the factorization process to handle varying levels of confidence in observed interactions.

Matrix Factorization: WRMF decomposes the user-item interaction matrix into two lower-dimensional matrices, one representing users and the other representing items.

Weighting Scheme: Each entry in the interaction matrix is associated with a weight, which reflects the confidence level in the interaction. Higher weights are assigned to observed interactions and lower weights to unobserved interactions.

Loss Function: The loss function in WRMF is modified to include the weights, ensuring that the factorization process pays more attention to interactions with higher confidence.

2.3 Bayesian Inference Variational Autoencoder(BIVAE)

Bayesian Inference Variational Autoencoder (BIVAE) is a deep learning-based model used for recommendation systems. It combines the principles of Bayesian inference with variational autoencoders to model user preferences and generate recommendations.

Variational Autoencoder (VAE): BIVAE builds on the VAE framework, which consists of an encoder and a decoder. The encoder maps user-item interactions to a latent space, and the decoder reconstructs the interactions from the latent representation.

Bayesian Inference: BIVAE incorporates Bayesian inference to model the uncertainty in the latent space. This allows the model to capture more robust representations of user preferences.

Probabilistic Modeling: The model uses a probabilistic approach to generate recommendations, accounting for the uncertainty and variability in user behavior.

3. Dataset Preprocessing

The datasets of this project are two csv files. Each file contains the users' ratings on the items. Since the two files are same in structure and I will use RatioSplit to split the dataset for validation and testing and I am aiming for recommending items for each user, I combined the dataset into one.

The final dataset is [281321 rows x 3 columns] and the first columns represents the user_id, the second column represents item_id an

d the last column is the ratings of the user on the item.

Then I used from_uir function (from cornac package) and itertuples function (from pandas' package) to transform the final dataset into a Dataset object for training and test with cornac. Finally, I get a Dataset with 21124 users and 31203 items.

In the training process, I utilized the RatioSplit to split the dataset into training(80%), validating(10%,20%) and testing(0,10%) datasets. The rating threshold is 4.0 which means only the ratings above 4.0 will be considered as positive interaction.

4. Experiment

During the experiment phase, I used GridSearch for hyper tuning on BIVAE and BPR model. For BIVAE model the grid as follows:

Latent dimension	50	100	200
Encoder structure	[50]	[100]	[50,100]
Learning rate	0.001	0.01	0.1
Batch size	100	200	300

The best parameter for BIVAE model is k=50,encoder structure = [100],learning rate =0.01, batch size = 100.

For BPR model the gird as follows:

K	50	100	
Max iteration	50	100	200
Learning rate	0.001	0.01	0.1
Lambda	0.001	0.01	0.1

The best parameter for BPR model is k =100, max iteration =200,learning rate = 0.1,lambda = 0.01.

For WRMF model, since the time was not enough for me to do the GridSearch, I changed the hyper-parameters manually, then I find the best parameter is k = 100,max iteration = 100, a =1,b=0.1,learning rate = 0.001 and lambda for user and item are both 0.01.

Following is the result of the three model:

Model	NCRR@50	NDCG@50	Recall@50
BIVAE	0.0164	0.0323	0.0912
BPR	0.0209	0.0438	0.1288
WMF	0.0545	0.0724	0.1329

We can see clearly that WMF is the best model!

5. Comparison

The experimental results indicate that WRMF is the most effective model for this dataset, likely due to its robust handling of explicit feedback and its ability to incorporate confidence weights. BPR, while useful in implicit feedback scenarios, was less suited to this explicit feedback dataset. BIVAE, with its sophisticated deep learning architecture, has the potential for strong performance but requires careful tuning and possibly more computational resources to achieve optimal results.