MIE1516 Winter 2018 Project Proposal: Bayesian Optimization for Active Learning in Recommender Systems

Soon Chee Loong, cheeloong.soon@mail.utoronto.ca

1 Motivation

Youtube, Amazon are examples of classic recommender systems that recommend items from a large inventory. This project focuses on a particular type of personalized recommendation where it is relatively cheap to receive immediate sequential feedback from users on given recommendations such as those found on Tinder or Spotify. The goal is to effectively leverage sequential feedback and recommend items for active learning to enable personalized recommendation for each user.

2 Project Goal

This project aims to effectively perform and evaluate active learning in recommendation system offline using a black-box Bayesian Optimization approach. Bayesian Optimization enables adaptability for existing recommender systems as it only relies on the rating matrix and timestamps.

3 Approaches

3.1 Recommendation Models

This project compares the effectiveness of different types of models. Latent models are relatively simple point estimate models that are simple to train. Bayesian models are relatively powerful full posterior models, but requires approximate inference such as Variational Inference or MCMC Sampling. Sequential Models are able to leverage temporal data in recommendation. Hybrid approaches are the most powerful models that can leverage both full posterior and temporal data.

We decided to analyze popular models such as:

- Point Estimate Models
  - Latent Model: Probabilistic Matrix Factorization [2]
  - (Future Work) Linear Model: SLIM/LREC
  - (Future Work) Neural Network Model: Auto-Encoder
- Posterior Estimate Models
  - Variational Bayesian Approach [1]
- Sequential Models
  - (Future Work) Sequential Model (temporal): Recurrent Neural Network with Gated Recurrent Unit
- Hybrid Models
  - (Future Work): Bayesian Recurrent Neural Network

As an example of a probabilistic graphical model.

- σ_v = standard deviation for item
- σ_u = standard deviation for user
- σ = standard deviation for rating prediction
- U_i = latent model for user i
- V_j = latent model for item j
- R_{ij} = rating estimate for user i, item j

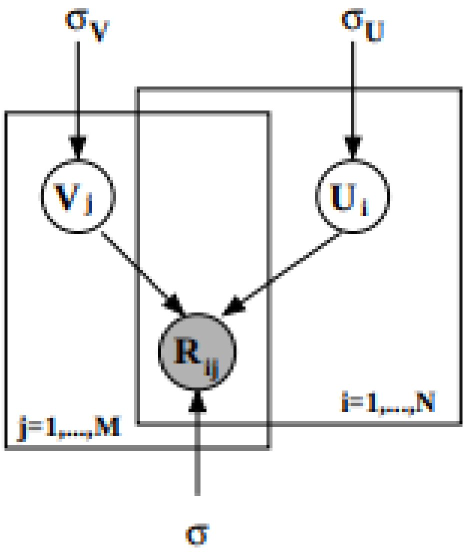

Figure 1: PMF Model: Taken from [2]

3.2 Posterior Estimation

It is known to be difficult to come up with a tractable closed-form posterior for our chosen model.

The different ways to estimate the posterior are:

- Histogram Density Estimation
- (Future Work) Kernel Density Estimation
- (Future Work) K Nearest Neighbour Density Estimation
- Moment Matching

3.3 Acquisition Function

We compare the various acquisition functions that leverages posterior of Bayesian modelling, and how they perform with respect to the various Estimation Posterior approaches. For models that do not estimate posterior, we simply train them using acquisition functions that doesn't require a posterior to evaluate such as epsilon-greedy.

- Greedy (only exploitation)
- Random (only exploration)
- epsilon-greedy
- Thompson Sampling
- Upper Confidence Bound
- Probability of Improvement
- Expected Improvement

4 Dataset

I used popular recommendation datasets that contains timestamps

- Movielens 1M
- Movielens 20M
- (Future Work) Netflix
- (Future Work) LastFM
- Spotify Million Dataset

5 Evaluation

To evaluate, the training data is ordered by timestamp, using the past as training data and future as test data. Training data is used to initialize the models to a state that achieves reasonable RMSE, but NOT personalized to any user. I save this initial non-personalized state for each model.

Each test user will be evaluated from the given initial state to be independent from other test user's data.

Then, I evaluate the model's active learning capability offline.

I partition the test data to a finite number of equal sized sets, that are sorted by time. Then, at each time interval, we pick best training point out of the pool. After that, we only train from all the training point out of the pool and evaluate on all training points in the next pool. This way, we ensure we are always only predicting the future and not cheating by predicting the past.

5.1 Evaluation: Acquisition Function

At each step, I select only one of the test data from the current set, based on the given acquisition function. This selected test data is evaluated using a regret approach, where the optimal rating is the highest rated test data in the current set that could have been selected.

Then, we the model is trained using either only the selected test data from the current set or all data points from the current set before it is repeated for the next set.

- Option1: Train on only the single selected data from current set
- Option2: Train on all data from current set

Option1 is motivated by immediate recommendation where each next recommendation is influenced by the previous recommendation feedback. However, it can be difficult for the model to significantly update based on only a single feedback.

Option2 is motivated by sequential recommendation that only updates after a given sequence of feedback. Option 2 also ensures that each step is independent of the choice the acquisition function made at the previous step to prevent double counting of errors. Option 2 is also motivated that it can be difficult for the model to learn from just an addition of a single point.

5.2 Evaluation: Model Adaptability

It is important for the model to be able to easily adapt to selected training points as an active learner. We train the saved initial model with a fixed number of iteration only the training data for each user separately. Then, we calculate the RMSE with respect to that user at each iteration and plot the RMSE for User vs Number of training iterations. This measures how effective and efficient the model is able to adapt to a specific user for personalization.

References

[1] Yew Jin Lim and Yee Whye Teh. Variational bayesian approach to movie rating prediction. In *Proceedings of KDD cup and workshop*, volume 7, pages 15–21, 2007.

[2] Andriy Mnih and Ruslan R Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.