Künstl Intell (2014) 28:329–332
DOI 10.1007/s13218-014-0323-2

DOCTORAL AND POSTDOCTORAL DISSERTATIONS

Active Learning for Recommender Systems

Rasoul Karimi

Published online: 23 August 2014

1 Introduction

Recommender systems learn user preferences and provide them personalized recommendations. Evidently, the performance of recommender systems depends on the amount of information that users provide regarding items, most often in the form of ratings. This problem is amplified for new users because they have not provided any rating, which impacts negatively on the quality of generated recommendations. This problem is called *new-user problem*. A simple and effective way to overcome this problem is posing queries to new users so that they express their preferences about selected items, e.g., by rating them. Nevertheless, the selection of items must take into consideration that users are not willing to answer a lot of such queries. To address this problem, *active learning* methods have been proposed to acquire the most informative ratings, i.e., ratings from users that will help most in determining their interests. Active learning is a learning algorithm that is able to interactively query the Oracle to obtain labels for data instances. The Oracle is a user or teacher who knows the labels.

The aim of this dissertation [8] is to take inspiration from the literature of active learning for classification (regression) problems and develop new methods for the new-user problem in recommender systems. In the recommender system context, new users play the role of the Oracle and provide ratings (labels) to items (data instances). Specifically, the following questions are addressed in this dissertation: (1) which recommendation model is suitable for active-learning purposes? (Sect. 2) (2) how can active learning criteria be adapted and customized for the new-user problem and which one is the best? (Sect. 3) (3) what are the specific requirements and properties of the new-user problem that do not exist in active learning and how can new active learning methods be developed based on these properties? (Sects. 4, 5).

2 Comparing Prediction Models

The accuracy of active learning methods heavily depends on the underlying prediction model of recommender systems. Therefore, we need to choose a right model in the first place. When I started to work on this dissertation, the state-of-the-art active learning methods for recommender systems were based on Aspect Model (AM) [3, 4]. The AM is a probabilistic latent model for the analysis of matrix or tensor data. However, recent research (especially as has been demonstrated during the Netflix[1] challenge) indicates that Matrix Factorization (MF) is a superior prediction model for recommender systems. The MF maps users and items into a latent space and then items that are in the neighborhood of the target user in the latent space are recommended to her. At first I compared MF to AM and showed that regardless of any active learning method, MF outperforms AM, both in terms of accuracy and time. Therefore, it is promising to develop active learning methods based on MF. Hence, in the rest of the dissertation, I used MF for active-learning purposes.

R. Karimi (✉)
Information Systems and Machine Learning Lab Marienburger Platz 22, University of Hildesheim, 31141 Hildesheim, Germany
e-mail: karimi@ismll.uni-hildesheim.de

[1] http://www.netflixprize.com/.

3 Active Learning for Full Oracle

In active learning, it is supposed that the Oracle has full rationality and is able to label any queried instance. Following this assumption, I supposed that the new users are full Oracles and can rate all queried items. Then, I adapted a couple of existing active learning criteria for the classification (regression) problem for the new-user problem. The developed methods are as follows:

- *Uncertainty in latent space* Uncertainty is a typical criterion for active learning for linear discriminant classifiers, such as SVM. Specifically, the uncertainty of instances that are close to the decision boundary is higher than examples that are far from the decision boundary [11]. I could get my inspiration from this definition and developed an uncertainty measure for MF.
- *Uncertainty in rating space* In recommender systems, there is a matrix of ratings of users to items. Based on these ratings, we can use the item average method to predict the ratings of all items for the new users. However, the predictions are not personalized. It means all new users will receive the same predictions. On the other hand, we can have a second prediction model that, based on a few ratings provided by new users, computes personalized predictions. The difference between these two predictions indicates how uncertain we are about the prediction of the personalized model. The higher the difference, the higher the uncertainty.
- *Non-myopic active learning* In contrast to myopic active learning methods that aim at finding the best *next* query, in the non-myopic approach the objective is to find the best *sequence* of queries [9]. The major consequence of this difference is that non-myopic active learning may choose an example that is not the best next query, but considering the next queries, chooses such that the sequence of all queries is optimal. I proposed a non-myopic active learning method that is based on the characteristics of MF. First, it explores the latent space to get closer to the optimal new user features. Then, it exploits the learned features and slightly adjusts them [5].
- *Optimal active learning* The principle of optimal active learning defines the optimal query as following: the optimal data instance for a query is the instance in which, when its label is received from the Oracle and is added to training data, retraining the prediction model with the new training data leads to the minimum error on the test data [1]. I applied this principle to MF and using some simplifications, a closed-form formula was derived that approximates optimal active learning for MF [6].

4 Factorized Decision Trees

When we apply an active learning technique to the new-user problem, it would be more realistic if we consider new users as partial Oracles instead of full Oracles. This means if they do not know the queried item, they simply do not rate it. For example, in a movie recommendation scenario, if the new users have not watched a movie, they do not rate it since they cannot evaluate it.

Decision trees have been proposed as the *query prediction* model for a partial Oracle [2]. In [2], ratings are predicted using the item average method, which is simple. On the other hand, the item average method is fast and keeps the learning algorithm tractable. Zhou et al. [12] proposed a solution to incorporate MF into the decision trees, however it is slow and not scalable. In my dissertation, I proposed a scalable approach to add MF to the decision trees. It is called Factorized Decision Trees (FDT). The FDT divides the learning algorithm of decision trees into two steps. First the structure of the trees is learned as in [2]. Then the rating predictions (labels) are updated by MF. In this way, we achieve a method that is scalable and accurate.

When the new user preference elicitation ends and a couple of ratings are received from the new user, she is treated as a normal user like existing users of the recommender system. On the other hand, there is already a recommendation model for the existing users, which is usually MF. We call it warm MF since it is for users who already have enough ratings in the data set, in contrast to cold (new) users who have a few ratings. Now we need to fill the gap between the new users and the existing users by folding the new user into the warm MF. Specifically, we need to learn the latent features of the new user in the warm MF. A naive approach for doing this is to add the ratings of the new user to the original data set and then retrain the MF with the whole training data set. However, as we have to repeat this process for all new users, it would be very slow. Therefore, we have to switch to online updating. In online updating, using the ratings that the new user has given, only the new user's latent features are updated and the rest of the features including item features and other user features are not touched [10].

Fortunately, the FDT can already provide us with the new user's latent features and there is no need to use online updating. The FDT generates user features for each type of new users, which corresponds to the leaf nodes of decision trees. In this way we learn the new user features with a higher accuracy. Moreover, there is no need for an online updating step to bridge the query prediction model (decision trees) and the recommendation model (MF). In fact, this is a new fold-in approach, in which, given a new user with a few ratings, a subset of training users who have the same ratings like the new user are selected and then the

new users features are trained using all ratings of these users. In my experiments, I found out that FDT can become even faster if only user features are updated and the rest of the features are fixed to the warm MF. However, the accuracy is slightly affected.

5 Learning Active Learning

Compared to the application of active learning in classification (regression), active learning in recommender systems presents several differences. The reason is that although there are no ratings for new users, there is an abundance of available ratings—collectively— from past (existing) users. In this dissertation, I proposed an innovative approach for active learning in recommender systems, which aims at taking advantage of this additional information. The main idea is to consider existing users as (artificial) new users and solve an active learning problem for each of them. In the end, we aggregate all solved problems in order to learn how to solve a active learning problem for a real new user. This is why this approach is called Learning Active Learning (LAL).

To make the LAL adaptive to the new user's responses, I used decision trees as it was proposed by [2] but with a minor difference. [2] opts for 3-way splits corresponding to three possible user responses ("Like", "Dislike", and "Unknown"). In datasets like Netflix and MovieLens where the range of ratings is from 1 to 5, ratings from 1 to 3 are considered as "Dislike" and ratings 4 and 5 are treated as "Like". Moreover, the missing ratings are considered as "Unknown", meaning users do not know the queried item, so they can not rate it. However, it is expected that a more refined split, such as a 6-way split that matches five star levels plus an "unknown" would improve accuracy

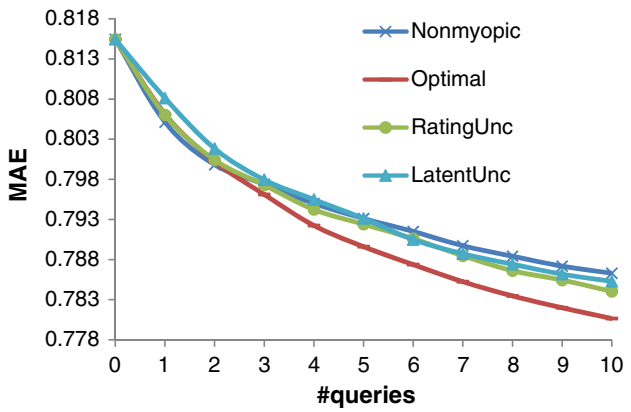

Fig. 1 MAE results of Optimal, Non-myopic, Uncertainty in the rating space (RatingUnc), and uncertainty in the latent space (LatentUnc)

because it distinguishes user tastes more precisely and in a more fine-grade fashion. The main bottleneck to do so is the overhead caused by increasing the number of nodes. The higher the number of splits, the higher the number of nodes, which requires more time to build the decision trees. To solve this issue, I proposed a sampling method that drastically speeds up the tree construction algorithm. This method is called Most Popular Sampling (MPS). Instead of checking all candidate items at each node, the MPS checks only those items that are most popular among users associated with the node. Given that MPS is used, the 6-way split can be leveraged to improve the accuracy of rating predictions while the tree learning algorithm is still tractable.

6 Experimental Results

In this section, first I report the results of the full Oracle and then the results of the partial Oracle is presented. For the full Oracle, I conducted the experiments on the Movielens (100k)[2] in 10 fold-cross validations and measured the error based on Mean Absolute Error (MAE).

Figure 1 shows the results of optimal, non-myopic, uncertainty in the rating space (RatingUnc) and uncertainty in the latent space (LatentUnc). I do not compare these methods against [3, 4] because they are based on AM, which is not a suitable model for active learning [5]. Non-myopic, RatingUnc and LatentUnc exhibit more or less the same behavior. They perform well in the first queries but at the end converge to the same point. In contrast, the optimal method outperforms other methods from the beginning to the end because it has a different approach. It aims to directly optimize the test error. That is why its performance continues and does not converge to the random selection. Therefore, if the new users are ready to provide more ratings, the optimal method can efficiently use them to improve the accuracy.

Now we go on to report the results of partial Oracle. For this experiment, I used the Netflix data set and measured the error based on Root Mean Square Error (RMSE). Figure 2 shows the results of FDT and LAL. For LAL, I investigated two different types of models: item average (LAL-Boot) and the second on MF (LAL-FDT). The main baseline is [2] which is called Boot in Figure 2. I also compare to three rather simple baselines. Random selects queries at random. Global Most Popular (GMP) selects items that are most popular in the data set. And Local Most Popular (LMP) selects items that are most popular based on the local ratings in each node of decision trees. As the results show, the three simple baselines perform worse than

[2] http://grouplens.org/datasets/movielens/.

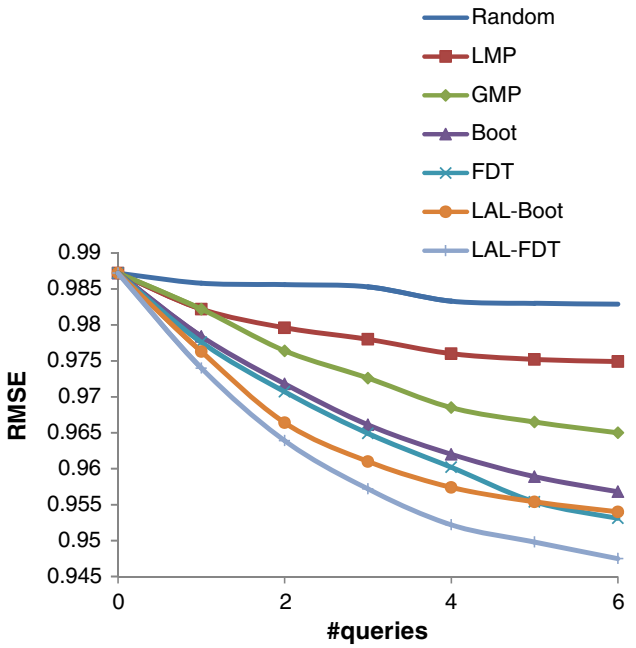

Fig. 2 RMSE results of three simple baselines, LAL-FDT, LAL-Boot, FDT and Bootstrapping

Table 1 The RMSE of online updating in MF and FDT after each query

	0	1	2	3	4	5
Online	1.056	1.0290	1.0056	1.0008	0.9948	0.9914
FDT	0.9872	0.9776	0.9707	0.9649	0.9602	0.9554

the main baseline (Boot). Also, the proposed approaches (FDT and LAL) outperform Boot, especially LAL-FDT.

I finish this section by comparing the FDT to online updating [10]. To use online updating, first decision trees are built as in [2]. Then in the leaf nodes, the user features are retrained only based on the received ratings from the root node to the leaf node. Table 1 reflects the RMSE after each query. Clearly FDT outperforms online updating by a large margin. This happens because FDT uses all ratings of the training users who are similar to the new user to train the user features. However, online updating uses only a few ratings that are received from the new user. The more the number of ratings, the better the accuracy of the learned user features. Interestingly, online updating is even worse than Boot [2], which is based on item average prediction. However, as the Boot method does not provide the latent features, it cannot be used to fold the new user into the warm MF model.

References

1. Cohn DA, Jordan M (1995) Active learning with statistical models. In: Proceeding of the advances in neural information processing systems (NIPS)
2. Golbandi N, Koren Y, Lempel L (2011) Adaptive bootstrapping of recommender systems using decision trees. In: Proceeding of the WSDM. ACM, New York
3. Harpale AS, Yang Y (2008) Personalized active learning for collaborative filtering. In: Proceedings of the 31st annual international ACM SIGIR
4. Jin R, Si L (2004) A bayesian approach toward active learning for collaborative filtering. In: Proceedings of the 20th conference on UAI
5. Karimi R, Freudenthaler C, Nanopoulosm A, Schmidt-Thieme, L (2011) Non-myopic active learning for recommender systems based on matrix factorization. In: Proceeding of the 12th IEEE international conference on information reuse and integration (IRI), Las Vegas, USA
6. Karimi R, Freudenthaler C, Nanopoulos A, Schmidt-Thieme L (2011) Towards optimal active learning for matrix factorization in recommender systems. In: Proceeding of the 23th IEEE international conference on tools with artificial intelligence (ICTAI), Florida, USA
7. Karimi R, Wistuba M, Nanopoulos A, Schmidt-Thieme L (2013) Factorized decision trees for active learning in recommender systems. In: Proceeding of the 25th IEEE international conference on tools with artificial intelligence (ICTAI), Washington DC, USA
8. Karimi R (2014) Active learning for recommender systems. Cuvillier Verlag, Germany
9. Osugi T, Kun D, Scott S (2005) Balancing exploration and exploitation: a new algorithm for active machine learning. In: IEEE international conference on data mining (ICDM)
10. Rendle S, Schmidt-Thieme L (2008) Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In: Proceeding of the ACM conference on recommender systems (RecSys)
11. Schohn G, Cohn D (2000) Less is more: active learning with support vector machines. In: Proceeding of the international conference on machine learning (ICML)
12. Zhou K, Yang SH, Zha H (2011) Functional matrix factorizations for cold-start recommendation. In: Proceedings of 34st annual international ACM SIGIR

Rasoul Karimi was born in 1980 in Tehran. He studied computer engineering and got his master's degree in 2005 from the University of Tehran. He started his PhD in 2009 in the University of Hildesheim and in 2014 he was awarded his PhD.