RankMBPR: Rank-aware Mutual Bayesian Personalized Ranking for Item Recommendation

Lu Yu[1], Ge Zhou[1], Chuxu Zhang[2], Junming Huang[3], Chuang Liu[1⋆], and Zi-Ke Zhang[1⋆]

[1]Alibaba Research Centre for Complexity Sciences, Hangzhou Normal University
[2]Department of Computer Science, Rutgers University
[3]Web Sciences Centre, University of Electronic Science and Technology of China
{coolluyu@gmail.com,flyzhouge@outlook.com,cz201@cs.rutgers.edu, mail@junminghuang.com,liuchuang@hznu.edu.cn,zhangzike@gmail.com}

Abstract. Previous works indicated that pairwise methods are state-of- the-art approaches to fit users' taste from implicit feedback. In this paper, we argue that constructing item pairwise samples for a fixed user is insufficient, because taste differences between two users with respect to a same item can not be explicitly distinguished. Moreover, the rank position of positive items are not used as a metric to measure the learning magnitude in the next step. Therefore, we firstly define a *confidence function* to dynamically control the learning step-size for updating model parameters. Sequently, we introduce a generic way to construct mutual pairwise loss from both users' and items' perspective. Instead of user-oriented pairwise sampling strategy alone, we incorporate item pairwise samples into a popular pairwise learning framework, *bayesian personalized ranking* (BPR), and propose *mutual bayesian personalized ranking* (MBPR) method. In addition, a rank-aware adaptively sampling strategy is proposed to come up with the final approach, called RankMBPR. Empirical studies are carried out on four real-world datasets, and experimental results in several metrics demonstrate the efficiency and effectiveness of our proposed method, comparing with other baseline algorithms.[1]

1 Introduction

Building predictor for top-k recommendation by mining users' preferences from implicit feedback [1] can help to produce recommendations in a wide range of applications [14, 9, 15, 17]. One significant challenge is that only positive observations are available. For example, we can only observe that a user bought a book, or a movie ticket from the web logs. Such issue is called "one-class" recommendation and many works have offered solutions to measure relevance of a user-item pair by transforming traditional rating estimation problem to a ranking task.

To solve the one-class recommendation problem, Rendle *et al.* [2] proposed *bayesian personalized ranking* (BPR), a popular pairwise learning framework.

⋆ Corresponding Author.

[1] The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-39937-9_19

Different from pointwise method dealing with a regression problem [5], BPR is built under assumption that everything that a user has not bought is of less interest for the user than those bought items. Empirically, pairwise ranking methods can perform better than pointwise approaches, and have been used as the workhorse by many recommendation approaches to tackle challenges from different applications, like tweet recommendation [22], tag recommendation [9], relation extraction [7], entity recommendation in heterogeneous information network [21] etc. In addition, many works aiming to optimise pairwise preference learning for one-class recommendation problems are recently proposed via leveraging novel information like social connection [19], group preference [8], or improving the strategy of selecting pairwise samples [3].

To the best of our knowledge, current pairwise learning methods [2, 19, 20, 8] construct pair examples from the user side. In this paper, we argue that unilaterally constructing samples from users' perspective is insufficient based on one main reason: taste differences between two users with respect to a same item can not be explicitly distinguished. Therefore, we introduce *mutual bayesian personalized ranking* (MBPR) to offer alternative idea to express the pairwise interactions, instead of the *user-based pairwise preference* alone. In addition, inspired by the recent literature [3], we optimize the sampling strategy of MBPR via utilising the rank position of positive samples to dynamically control the learning speed. Experimental results on four real-world datasets show that our proposed method significantly improves the performances comparing with other baseline algorithms on four evaluation metrics.

2 Preliminary

Let $\mathcal{U}$ and $\mathcal{I}$ denote the user and item set, respectively. We use $\mathcal{T}$ to denote the observed feedbacks from n users and m items. Each case $(u, i) \in \mathcal{T}$ means that user u ever clicked or examined item i. For a given user u, the relationship between user u and item i can be measured as x_{ui}, then a recommendation list of items could be generated from items $\mathcal{I} \setminus \mathcal{I}_u$, where $\mathcal{I}_u$ denotes the clicked or examined items by user u. In practise, only top-k recommendations can attract users' attention, and recommendation task in such a case can be modified as a learning to rank problem. In order to represent user u' preference over items with only "one-class" observations $\mathcal{T}$, pairwise learning approaches typically regard observed user-item pairs $(u, i) \in \mathcal{T}$ as a positive class label, and all other combinations $(u, j) \in (\mathcal{U} \times \mathcal{I} \setminus \mathcal{T})$ as a negative one. Then intermedia training samples $(u, i, j) \in D_{\mathcal{T}}$ are constructed according to assumption 1.

Assumption 1 *For a given user u, unequal relationship exists between the examined item i and the unexamined item j, and user u would show more preference to item i than item j.*

2.1 Bayesian Personalised Ranking (BPR)

For a given user u, the relationship between user u and item i can be measured as x_{ui}. BPR will fit to the dataset $D_{\mathcal{T}}$ to correctly learn the ranks of all items

via maximizing the following posterior probability of the parameter space θ.

$$p(\theta| >_u) \propto p(>_u |\theta)p(\theta), \tag{1}$$

where notation $>_u = \{i >_u j : (u,i,j) \in D_{\mathcal{T}}\}$ denotes the pairwise ranking structure for a given u, and $p(\theta)$ is the prior probability of parameter space θ. Then BPR assumes that each case of $>_u$ is independent, and the above likelihood of pairwise preferences (LPP) can be modified as $LPP(u) = \prod_{(u,i,j)\in D_{\mathcal{T}}} p(i >_u j|\theta)$, where $p(i >_u j|\theta) = \sigma(x_{uij}(\theta))$, and $\sigma(x) = \frac{1}{1+e^{-x}}$. In terms of $p(\theta)$, we define it as a normal distribution with zero mean and covariance matrix $\sum_\theta = \lambda_\theta I$, that is, $\theta \backsim \mathcal{N}(\mathbf{0}, \sum_\theta)$. Now we can infer the BPR by filling $p(\theta)$ into the maximum posterior probability in Equation (1).

$$\ln LPP(u) + \ln p(\theta) = \sum_{(u,i,j)\in D_{\mathcal{T}}} \ln \sigma(x_{uij}) - \frac{\lambda_\theta}{2}||\theta||^2, \tag{2}$$

where λ_θ are model regularization parameters. Here we choose $x_{uij}(\theta) = x_{ui} - x_{uj}$. The specific definition of the preference function used in this paper is $x_{ui} = b_u + b_i + W_u V_i^\top$, where $W \in \mathbb{R}^{|\mathcal{U}|\times d}, V \in \mathbb{R}^{|\mathcal{I}|\times d}$, and d is the number of latent factors. b_u and b_i are bias features for each user u and for each item i, respectively. Therefore, $x_{uij}(\theta) = b_i + W_u V_i^\top - b_j - W_u V_j^\top$. Generally, stochastic gradient descent based algorithms can be used as a workhorse to optimise the posterior distribution. More specifically, the parameters θ are randomly initialized. With iteratively traversing each observation $(u,i) \in \mathcal{T}$, a negative sample j is picked and parameters θ can be updated with gradients as shown in Equation (3) until the stopping criterion is matched, and return the learned model.

$$\theta \leftarrow \theta + \alpha(\frac{\partial \ln LPP(u)}{\partial \sigma(x)} \cdot \frac{\partial x_{uij}}{\partial \theta} - \lambda_\theta \theta) \tag{3}$$

According to literature [2], the way to select a negative j could have significant impact on the performance of BPR. Typically, a bootstrap sampling approach with replacement is suggested, *i.e.* for a given (u,i), j is randomly selected out depending on an uniform distribution. However, Rendle *et al.* [3] argued that for a given user u, good negative samples which could make an effective updating could lie in a higher position than positive item i. While, uniform sampling could equally regard each negative item without taking the ranks of positive items i into consideration. Sequently, Rendle *et al.* [3] proposed a novel adaptive sampling strategy after giving insight into the learning diagram, especially the evolution of gradient distribution. Series of experiment results demonstrate that adaptive sampling a negative j according to their position in the rank list could improve the convergence speed and effectiveness of BPR. In more detail, the adaptive sampling strategy is described as follows:
For sampling a negative item j for the given (u,i), we firstly extend both user's and items' latent feature, that is, $W'_u = [1, W_u], V'_i = [b_i, V_i]$.

1. Sample a rank r from distribution $p(r) \propto \exp(\frac{-r}{\gamma_i})$, where γ_i is a hyperparameter controlling the expectation position of sampled item j. A small value

of γ_i could generate a small value of r with a high probability. It indicates that a negative item j ranking in a high position could be probably sampled.
2. Sample a factor dimension d according to probability distribution $p(d|u) \propto |W'_{ud}|\delta_d$, where δ_d denotes the variance over all items' dth factor, and W'_{ud} denotes the value of user u's dth factor.
3. Sort items according to $V'_{\cdot,d}$, and return the item j on position r in the sorted list.

However, as we see, about $|\mathcal{I}\backslash\mathcal{I}_u|$ negative cases need be examined for each given (u, i). If the number of items in the database is enormous, the scale of negative candidates make it intractable to directly apply such strategy. In order to reduce the complexity, Rendle *et al.* proposed to pre-compute the ranks every $|\mathcal{I}|\log|\mathcal{I}|$ training cases.

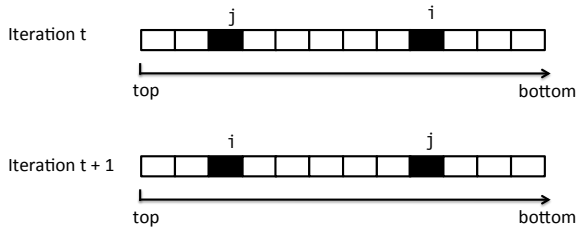

Fig. 1: Simple illustration to describe the item permutation for user u in different learning stages. The term "top" and "bottom" stands for the top and bottom position in the rank list, respectively.

3 Our Solution

3.1 Rank-aware Pairwise Learning

In last Section 2.1, we can see that adaptively ranking the items could be implemented every a fixed steps in order to abate the computation cost. However, we argue that it still needs to sort $|\mathcal{I} \setminus \mathcal{I}_u|$ items for each $u \in \mathcal{U}$, which will dramatically increase the computation resources with the growth of the size of user and item set. Moreover, the rank position of the positive item i is not utilized as a significant criterion to measure the learning performance. One basic assumption of adaptive sampling is that selecting the negative sample j ranking in a higher position could help to find a good case to effectively update pairwise parameters. However, it omits that the ranks of positive item i when utilizing the benefits brought by the position information of negative item j. We believe that utilising the rank information of positive item i is crucial to offer an alternative way to improve the performance of pairwise learning algorithm based on BPR.

Let π_t^u denote the item permutation of user u in the iteration t, and $\pi_t^u(i)$ represents the position of item i. Suppose that for a given $(u, i) \in \mathcal{T}$ in iteration t, $\pi_t^u(i)$ is very close to the bottom as it is shown Figure 1. At that moment, user u's preference over positive item i is insufficiently learned under this situation. It still has a huge position gap between item i and j. If we sample the negative j which

ranks much higher than a positive item i, we should update the corresponding parameters with a large magnitude due to the significant position gap. If positive item i ranks closely to the top position in iteration $t+1$, we could say that preference function x_{ui} is learned well for this case, then we should update parameters with a slight step size. Therefore, we propose to transform the rank information as a crucial criterion to support the updating magnitude of model parameters.

To tackle the aforementioned challenge, we propose *rank-aware bayesian personalized ranking* (RankBPR) to utilize the rank position of item i for a given (u, i). We define a *confidence function* $C_u(rank_i)$, which serves as dynamic weight to control the updating step size of model parameters. Generally, $C_u(rank_i)$ is a monotonically increasing function. By integrating $C_u(rank_i)$ into Equation (2), we obtain the *rank-aware bayesian personalised ranking* (RankBPR), and the objective function can be modified as follows:

$$\ln LPP(u) + \ln p(\theta) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}_u} C_u(rank_i) \sum_{j \in \mathcal{I} \setminus \mathcal{I}_u} \ln \sigma(x_{uij}) - \lambda_\theta ||\theta||^2. \tag{4}$$

Correspondingly, the flow of stochastic gradient as shown in Equation (3) could be revised as follows:

$$\theta \leftarrow \theta + \alpha(C_u(rank_i) \cdot \frac{\partial \ln LPP(u)}{\partial \sigma(x)} \cdot \frac{\partial x_{uij}}{\partial \theta} - \lambda_\theta \theta) \tag{5}$$

As we see, the key part of the *confidence function* lies in the rank position of item i. In this work, we define $C_u(rank_i)$ as follows:

$$\begin{aligned} C_u(rank_i) &= \sum_{s=1}^{rank_i} \gamma_s, with\ \gamma_1 \geq \gamma_2 \geq \gamma_3 \geq ... \geq 0. \\ rank_i &= \sum_{j \in \mathcal{I} \setminus \mathcal{I}_u} \mathbb{I}[\beta + \hat{h}_{uj} \geq \hat{h}_{ui}] \end{aligned} \tag{6}$$

where $\mathbb{I}$ is the indicator function, and $\hat{h}_{ui}$ measures the relevance between user u and item i. β is a margin parameter to control the gap between the positive item i and negative sample j. We choose $\gamma_s = 1/s$ as the weighting approach, which could assign large value to top positions with rapidly decaying weight for lower positions. Intuitively, a positive item i, which ranks in the highest position, could produce a low confidence to update the parameters. Oppositely, a bottom positive item i could offer a high confidence. In terms of the $\hat{h}_{ui}$, we give two different definitions under the assumption that ranks of a item list can be measured as the inner-product of user-item features or only according to a specific dimension of users' feature vector. For simplicity, one can denote $\hat{h}_{ui} = x_{ui}$ as a response to measure the relevance of user u and items i as the product of their feature vectors.

$$\hat{h}_{ui} = b_u + b_i + W_u V_i^\top \tag{7}$$

Besides this, we suppose that the value of each element of user u's feature vector W_u can represent u's specific taste, which drives us to rank all items only

depending on the match score in a single dimension of feature vectors. To formulate our idea, we define a probability distribution for sampling a factor d as $p(d|u) \propto |W_{ud}|$. For each training case (u, i), we firstly sample a factor d according to $p(d|u) \propto |W_{ud}|$, then calculate $rank_i$ based on the match score $\hat{h}_{ui}$ between a user-item pair:

$$\hat{h}_{ui} = W_{ud}V_{id}, \tag{8}$$

where $W_{\cdot d}$ and $V_{\cdot d}$ denotes the value of dth element.

3.2 Fast Estimating Rank Position of Item i

Note that in each iteration, the exact value of $rank_i$ needs to be calculated for each observation is extremely expensive, when there are massive amount of items in the database. In this work, we employ a sampling process [10] to fast estimate the $rank_i$. More specifically, for a given sample (u, i), one uniformly draws a random negative item from $\mathcal{I}$ until finding a j, which satisfies $\beta + \hat{h}_{uj} \geq \hat{h}_{ui}$. Then the $rank_i$ can be approximated as $rank_i \approx \lfloor \frac{|\mathcal{I}|-1}{K} \rfloor$, where $\lfloor \cdot \rfloor$ denotes the floor function and K is the number of steps to find a item j. From this approximation of $rank_i$, we can see that the computations on seeking a negative item j could increase sharply if the positive item i happens to rank at the top of the list. To accelerate the estimation of $rank_i$, we define a parameter κ to limit the maximum sampling steps. Correspondingly, we also utilise an item buffer $buffer_{ui}$ whose size equals to κ to store every sampled negative item j. Finally, $rank_i$ can be approximated as $rank_i \approx \lfloor \frac{|\mathcal{I}|-1}{min(K,\kappa)} \rfloor$, and the negative item j will be selected from the top of the sorted $buffer_{ui}$ in descending order based on $\hat{h}_{uj}$. The complete procedure of RankBPR can be described in Algorithm 1.

Algorithm 1 Optimizing models with RankBPR

Input: $\mathcal{T}$: Training Set
Output: Learned $\hat{\theta}$

1: initialize θ
2: **repeat**
3: randomly draw (u,i) from $\mathcal{T}$
4: estimate $C_u(rank_i)$ according to Equation (6)
5: select a negative sample j from the top of the sorted $buffer_{ui}$
6: $x_{uij} \leftarrow x_{ui} - x_{uj}$
7: $\theta \leftarrow \theta + \alpha(C_u(rank_i) \cdot (1 - \sigma(x_{uij})) \frac{\partial}{\partial \theta} x_{uij} - \lambda_\theta \theta)$
8: **until** convergence.

3.3 Mutual Sample Construction for Pairwise Learning

Beside the optimized sampling strategy, we can see that pairwise learning approaches like BPR only focus on constructing samples (u, i, j) for a fixed user u. In this paper, we argue that only sampling negative samples w.r.t. a fixed user u is insufficient based on the following consideration:

– Recommendation task naturally involves with two basic types of entities, i.e. users and items. To better represent the relevance between them, mutual pairwise sampling is essential to not only capture a user's preferences over two different items, but also measure how different is a pair of users with respect to a same item. Intuitively, a piece of training sample (u, i, j) could help to distinguish the difference between item i and j from a user u's perspective, while two different users' preferences over the same item i could not be explicitly represented.

In order to tackle the aforementioned challenges, we propose to construct the pairwise loss from item's perspective instead of the construction of pairwise training samples from user's perspective alone. Specifically, another type of training samples $(i, u, v) \in D_{\mathcal{T}}$ for a given $(u, i) \in \mathcal{T}$ would be extracted under the proposed assumption 2.

Assumption 2 *A group of users who ever selected the same items might have closer tastes than other users. For a given item $(u, i) \in \mathcal{T}$, user u could have a stronger preference than user v, who did not ever explicitly click or rate item i.*

We think that users purchasing the same item i could have much closer connection than those who do not purchase item i. Following the assumption 2, mutual sampling construction could explicitly leverage the user connection information to distinguish the differences among users. Then two types of pairwise relationship are extracted for a given observation $(u, i) \in \mathcal{T}$. To formulate our idea, we propose *mutual bayesian personalized ranking* (MBPR) to incorporate mutual pairwise samples into BPR framework. As a response to our assumption, the basic idea of mutual pairwise sampling equals to user u- and item i-central pairwise construction. Therefore, two kinds of pairwise likelihood preferences, i.e. $p(i >_u j|\theta)$ and $p(u >_i v|\theta)$, are instantiated. The modified objective function we are going to maximise in this work is described as follows:

$$\begin{aligned} & \ln LPP(u) + \ln LPP(i) + \ln p(\theta) \\ = & \sum_{(u,i,j)\in D_{\mathcal{T}}} \ln \sigma(x_{uij}) + \sum_{(i,u,v)\in D_{\mathcal{T}}} \ln \sigma(x_{iuv}) - \frac{\lambda_\theta}{2}||\theta||^2. \end{aligned} \tag{9}$$

Different from standard BPR, mutual pairwise samples are constructed. In MBPR, for each observation $(u, i) \in \mathcal{T}$, a negative item $j \in \mathcal{I} \setminus \mathcal{I}_u$ and user $v \in \mathcal{U} \setminus \mathcal{U}_i$ are picked and parameters θ can be updated with the following gradients

$$\theta \leftarrow \theta + \alpha(\frac{\partial \ln LPP(u)}{\partial \sigma(x_{uij})} \cdot \frac{\partial x_{uij}}{\partial \theta} + \frac{\partial \ln LPP(i)}{\partial \sigma(x_{iuv})} \cdot \frac{\partial x_{iuv}}{\partial \theta} - \lambda_\theta \theta) \tag{10}$$

where $\mathcal{U}_i$ denotes the set of users who ever clicked item i. By employing rank-aware sampling approach, we further apply mutually dynamic sampling strategy to optimise the procedure of MBPR on selecting the negative samples and propose RankMBPR, in which parameters θ would be updated with the modified gradients:

$$\theta \leftarrow \theta + \alpha(C_u(rank_i)\frac{\partial \ln LPP(u)}{\partial \sigma(x_{uij})} \cdot \frac{\partial x_{uij}}{\partial \theta} + C_i(rank_u)\frac{\partial \ln LPP(i)}{\partial \sigma(x_{iuv})} \cdot \frac{\partial x_{iuv}}{\partial \theta} - \lambda_\theta \theta), \tag{11}$$

where $C_i(rank_u)$ can be calculated in the same way as $C_u(rank_i)$ with a slightly different definition as follows:

$$\begin{aligned} C_i(rank_u) &= \sum_{s=1}^{rank_u} \gamma_s, with\ \gamma_1 \geq \gamma_2 \geq \gamma_3 \geq ... \geq 0 \\ rank_u &= \sum_{v \in \mathcal{U} \setminus \mathcal{U}_i} \mathbb{I}[\beta + \hat{h}_{iv} \geq \hat{h}_{iu}]. \end{aligned} \tag{12}$$

Datasets	#Users	#Items	#Observations	#Density
ML100K	943	1682	100000	6.3%
Last.fm	1892	17632	92834	0.28%
Yelp	16826	14902	245109	0.097%
Epinions	49289	139738	664823	0.02%

Table 1: Statistics of the datasets

4 Experiments

Datasets & Settings: Four datasets[2] are used in this paper and statistics of them are summarized in Table 1. In this work, we adopt 5-fold cross validation method to demonstrate the performance of our proposed approach. More specifically, repeated validation experiments are conducted 5 times, in which we randomly select 80% of observed feedback as training set to train the ranking model, and the rest as the testing set. The final performance of each algorithm is measured as the average results on four top-N metrics, which are used to compare the performance in measuring recall ratio (Recall@5, Recall@10), precision of top-10 recommendation list (MAP@10, MRR@10). We mainly compare our method with the following approaches:

- **PopRec**: A naive baseline that generates a ranked list of all items with respect to their popularity, represented by the number of users who ever rated, or clicked the target items.
- **WRMF**: This method defines a weight distribution for each $(u, i) \in \mathcal{U} \times \mathcal{I}$, then employs matrix factorization model to solve a regression problem via optimizing a square loss function [5]. It is the state-of-the-art one-class collaborative filtering method.
- **ICF**: Item-based CF is a classical collaborative filtering approach, and was initially explored in the literature [13]. It is a generic approach which could be used for rating prediction and item recommendation with only implicit feedback.
- **BPR**: This method uses a basic matrix factorization model as the scoring function and BPR as the workhorse to learn the ranks of items [2].
- **AdaBPR**: This method is slightly different from BPR-MF as an adaptive sampling strategy is proposed to improve the learning efficiency of BPR [3]

[2] http://grouplens.org/datasets/movielens/
http://grouplens.org/datasets/hetrec-2011/
http://www.yelp.com/dataset_challenge
http://www.trustlet.org/wiki/Epinions

– **StaticBPR**: This method replaces the uniform sampling strategy with a static sampling strategy according to the popularity of items [3].

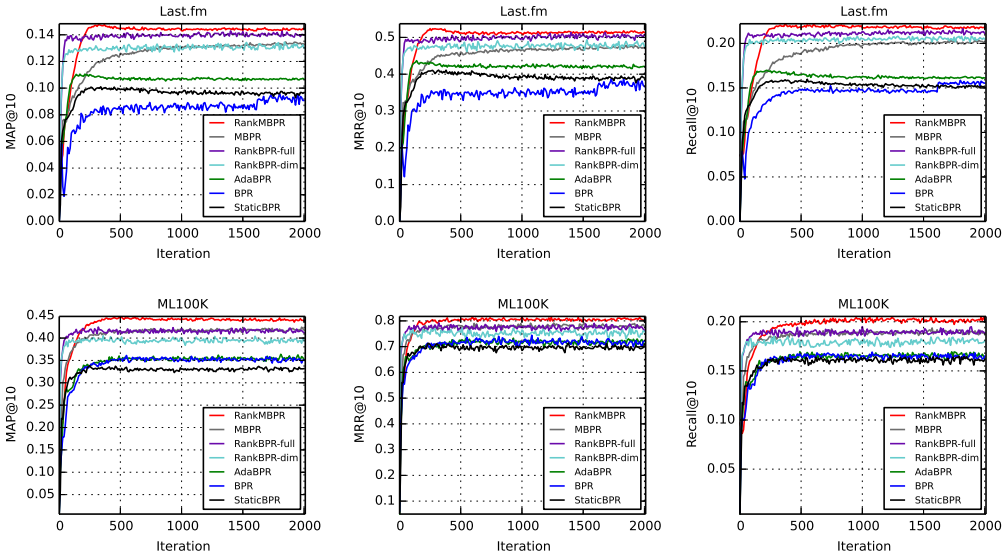

Fig. 2: Learning curves of different BPR-based algorithms in several evaluation metrics. In this case, we set dimension d to 30, and each algorithms iterated 2000 times. RankBPR-full takes Equation (7) as the relevance function for RankBPR algorithm, while RankBPR-dim employs Equation (8). In this work, we set $\beta = 1.0$ for both RankBPR-full and RankMBPR, and $\beta = 0.1$ for RankBPR-dim. Maximum size κ of $buffer_{ui}$ is fixed as 200 for RankBPR-full, RankBPR-dim, and RankMBPR.

4.1 Performance Evaluation

Convergence: Initially, we empirically study the learning curves of BPR-based algorithms with different pairwise sampling strategies (see Figure 2) on four datasets. Due to the limited space, we only list the learning curves on Last.fm and ML100K. From Figure 2, we can see that all algorithms almost converge after a number of iterations, then fluctuate in a tiny range around the converging performance in the different metrics. We can find that RankBPR is superior to other baselines, and RankBPR-full outperforms RankBPR-dim, which proves the efficiency and effectiveness of rank-aware sampling strategy with definition of $\hat{h}_{u,i}$ in Equation (7). Therefore, the way to calculate $\hat{h}_{u,i}$ for RankMBPR follows RankBPR-full. Turn to MBPR, its performance indicates that mutual construction of pairwise samples could also benefit for the pairwise learning methods. As incorporating both features of MBPR and RankBPR, RankMBPR achieves the best performance. From the learning curves we can see that stopping criterion could be defined as a fixed number of iterations for BPR-based algorithms. In this work, we set the number of iterations as 2000 for all BPR-based algorithms.
Performance Evaluation: Table 2 shows the average recommendation performance of different algorithms. We highlight the results of the best baseline and our proposed method respectively. Validation results show that RankMBPR is

superior to all baseline algorithms on all four datasets. Note that MBPR also has comparative better performance than baseline approaches. One possible reason may be that AdaBPR pays less attention to deeply explore the effects brought by the rank position of positive samples. In terms of WRMF, it is not directly to optimise ranking. Comparing with BPR, our proposed methods both utilise the differences among users by leveraging the construction of mutual pairwise samples, instead of the user-side solely, and the benefits brought by the rank-aware sampling method. From this table, we could find some interesting evidences. As the density of the dataset decreases, the performance gap between our proposed method and BPR-based methods will become larger. In particularly, AdaBPR performs not so much as ICF in the most sparse dataset, Epinions.

Datasets	Metrics	PopRec	ICF	WRMF	BPR	AdaBPR	MBPR	RankMBPR	Improv.
ML100K	Recall@5	0.0553	0.0882	0.1052	0.1037	**0.1123**	0.1142	**0.1202***	7.03%
	Recall@10	0.0974	0.1473	0.1764	0.1708	**0.1823**	0.1876	**0.1921***	5.37%
	MAP@10	0.1986	0.3041	0.3591	0.3712	**0.4024**	0.4188	**0.4312***	7.38%
	MRR@10	0.5449	0.6843	0.7236	0.7389	**0.7682**	0.7788	**0.7963***	4.39%
Last.fm	Recall@5	0.0476	0.1057	0.112	0.124	**0.133**	0.131	**0.147***	10.52%
	Recall@10	0.0747	0.1546	0.169	0.191	**0.202**	0.201	**0.218***	7.92%
	MAP@10	0.0413	0.1016	0.106	0.117	**0.128**	0.133	**0.145***	13.28%
	MRR@10	0.1995	0.4260	0.431	0.452	**0.482**	0.471	**0.512***	6.22%
Yelp	Recall@5	0.0156	0.0273	0.0262	0.0352	**0.0364**	0.0386	**0.0406***	11.53%
	Recall@10	0.0289	0.0471	0.0428	0.0596	**0.0601**	0.0638	**0.0672***	11.81%
	MAP@10	0.0098	0.0174	0.0172	0.0240	**0.0242**	0.0279	**0.0281***	16.11%
	MRR@10	0.0286	0.0459	0.0498	0.0640	**0.0648**	0.0724	**0.0742***	14.51%
Epinions	Recall@5	0.0143	**0.0293**	0.0227	0.0248	0.0261	0.0302	**0.0347***	18.43%
	Recall@10	0.0217	**0.0459**	0.0369	0.0416	0.0441	0.0487	**0.0548***	19.39%
	MAP@10	0.0097	**0.0199**	0.0159	0.0168	0.0178	0.0205	**0.0258***	29.64%
	MRR@10	0.0351	**0.0617**	0.0547	0.0518	0.0542	0.0628	**0.0802***	29.98%

Table 2: Experimental results of all baseline algorithms. The last column shows the improvement of the proposed method compared with the best baseline method. The latent dimension is fixed as $d = 30$ for MF-based methods, like WRMF, BPR, AdaBPR.

5 Related Work

Recently many works have began to adopt learning-to-rank idea to explore users' preference on items from ranking perspective [6, 16, 3, 2, 19, **?**,**?**]. As one of typical pairwise learning method, BPR is flexible to incorporate different contextual information to make BPR adaptive to different tasks. Riedel *et al.* [7] employs BPR to automatically extract structured and instructed relations. Zhao *et al.* [19] indicated out that users tend to have many common interests with their social friends, and proposed to leverage social connections to improve the quality of item recommendation. Pan *et al.* [8] pointed out that group features should be taken into account for further exploring users' preferences on items. Rendle *et al.* [9] extended BPR to learn tensor factorization method for recommending related tags to users with respect to a given item. Besides BPR, various methods

inherit the pairwise learning idea. In [10], Jason *et al.* defined order pairwise ranking loss and develop online Weighted Approximate-Rank Pairwise (WARP) method, which can be applied to various top-k learning problems. Later, Jason *et al.* in [11, 12] presented the possible applications of WARP in video recommendation and collaborative retrieval task. Zhao *et al.* [20] proposed a novel personalized feature projection method to model users' preferences over items. A boosting algorithm is also proposed to build ensemble BPR for one-class recommendation task.

6 Conclusions and Future Work

In this paper, we propose to complementarily distinguish users' taste differences from items' side. Two kinds of pairwise preference likelihood are defined. An improved sampling strategy is customized to our proposed method (MBPR), then an rank-aware MBPR (RankMBPR) is introduced to help sufficiently learn users' preferences over items. The experimental results on four datasets show that RankMBPR perseveres the most efficacy than all baseline algorithms. In future, we incline to further explore the probable effects of the contextual information, or the network structure on helping to select the good pairwise samples.

Acknowledgments

Chuxu Zhang thanks to the assistantship of Computer Science Department of Rutgers University. This work was partially supported by the National Natural Science Foundation of China (No. 11305043), and the Zhejiang Provincial Natural Science Foundation of China (No. LY14A050001), the EU FP7 Grant 611272 (project GROWTHCOM) and Zhejiang Provincial Qianjiang Talents Project (Grant No. QJC1302001).

References

1. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 426-434. ACM (2008)
2. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp. 452-461. AUAI Press (2009)
3. Rendle, S., Freudenthaler, C.: Improving pairwise learning for item recommendation from implicit feedback. In Proceedings of the 7th ACM International Conference on Web Search and Data Mining, pp. 273-282. ACM (2014)
4. Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., Yang, Q.: One-class collaborative filtering. In: Eighth IEEE International Conference on Data Mining, pp. 502-511. IEEE (2008)
5. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Eighth IEEE International Conference on Data Mining, pp. 263-272. IEEE (2008)
6. Shi, Y., Larson, M., Hanjalic, A.: List-wise learning to rank with matrix factorization for collaborative filtering. In Proceedings of the fourth ACM Conference on Recommender Systems, pp.269-272. ACM (2010)

7. Sebastian, R., Limin, Y., Andrew M.: Relation Extraction with Matrix Factorization and Universal Schemas. Joint Human Language Technology Conference/Annual Meeting of the North Computational Linguistics (HLT-NAACL), 2013
8. Pan, W., Chen, L.: GBPR: Group preference based bayesian personalized ranking for one-class collaborative filtering. In Proceedings of the Twenty-Third international joint conference on Artificial Intelligence, pp.2691–2697, IJCAI (2013)
9. Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization for personalized tag recommendation. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp.81-90, ACM (2010)
10. Weston, J., Bengio, S., Usunier, N.: Large scale image annotation: learning to rank with joint word-image embeddings. Machine Learning, 81(1), 21-35, 2010.
11. Weston, J., Yee, H., Weiss, R. J.: Learning to rank recommendations with the k-order statistic loss. In Proceedings of the 7th ACM Conference on Recommender Systems, pp.245-248, ACM (2013)
12. Weston, J., Wang, C., Weiss R., Berenzweig A.: In Proceedings of the 29th International Conference on Machine Learning, pp.9-16, ACM (2012)
13. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In Proceedings of International Conference on the World Wide Web, pp.285–295, ACM (2001)
14. Celma, O.: Music Recommendation and Discovery in the Long Tail. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.
15. Das, A. S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web, pp.271-280, ACM (2007)
16. Qiu ,H., Zhang, C., Miao, J.: Pairwise One Class Recommendation Algorithm, Advances in Knowledge Discovery and Data Mining Springer International Publishing, (9078) pp.744-755 (2015)
17. Qu, M., Qiu, G., He, X., Zhang, C., Wu, H., Bu, J., Chen, C.: In Probabilistic question recommendation for question answering communities. In Proceedings of the 18th international conference on World wide web, pp.1229-1230, ACM (2009)
18. Liu, Y., Zhao, P., Sun, A., Miao, C.: A Boosting Algorithm for Item Recommendation with Implicit Feedback. In Proceedings of International Joint Conference on Artificial Intelligence, IJCAI (2015)
19. Zhao, T., Mcauley, J., King, I.: Leveraging Social Connections to Improve Personalized Ranking for Collaborative Filtering. In Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, pp.261-270, ACM (2014)
20. Zhao T., McAuley, J., King I.: Improving latent factor models via personalized feature projection for one-class recommendation. In Proceedings of the 24th International Conference on Information and Knowledge Management, ACM (2015)
21. Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick B., Han, J.: Personalized entity recommendation: A heterogeneous information network approach. In Proceedings of the 7th International conference on Web search and data mining, pp. 283-292, ACM (2014)
22. Chen, K., Chen, T., Zheng, G., Jin, O., Yao, E., Yu, Y.: Collaborative personalized tweet recommendation. In Proceedings of the 35th international SIGIR conference on Research and development in information retrieval, pp. 661-670, ACM (2012)
23. Chen C., Yin H., Yao J., Cui B.: TeRec: A Temporal Recommender System Over Tweet Stream. In Proceedings of the 39th International Conference on Very Large Data Bases, pp.1254-1257, ACM (2012)
24. Yin H., Sun Y., Cui B., Hu Z., Chen L.: LCARS: A Location-Content-Aware Recommender System. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 221-229. ACM (2013)