Alternating Pointwise-Pairwise Learning for Personalized Item Ranking

Yu Lei, Wenjie Li, Ziyu Lu, Miao Zhao
Department of Computing, The Hong Kong Polytechnic University
Hong Kong
{csylei,cswjli,csmiaozhao}@comp.polyu.edu.hk;luziyuhku@gmail.com

ABSTRACT

Pointwise and pairwise collaborative ranking are two major classes of algorithms for personalized item ranking. This paper proposes a novel joint learning method named alternating pointwise-pairwise learning (APPL) to improve ranking performance. APPL combines the ideas of both pointwise and pairwise learning, and is able to produce a more effective prediction model. The extensive experiments with both explicit and implicit feedback settings on four real-world datasets demonstrate that APPL performs significantly better than the state-of-the-art methods.

CCS CONCEPTS

• **Information systems → Collaborative filtering**; **Learning to rank**; **Recommender systems**;

KEYWORDS

Personalized item ranking; collaborative ranking; item recommendation

1 INTRODUCTION

Personalized item ranking is a major task in recommender systems. The existing ranking algorithms proposed in the literature can be mainly grouped into two classes: collaborative filtering (CF) and collaborative ranking (CR). The basic idea of CF is to leverage user-item feedback data to infer user preferences, while CR can be regarded as a combination of the CF techniques and the learning to rank (LTR) techniques [9]. In particular, model-based CF methods [4–6, 8, 15, 16] seek to train a prediction model from feedback data by employing the techniques of matrix factorization (MF) where the scoring function is defined as the inner product of low-dimensional user and item feature vectors, and is learnt by minimizing certain pointwise loss functions. By contrast, CR methods [1, 2, 7, 10–14, 17] aim to directly optimize the ranking over feedback data by combining the MF scoring function from CF and various pairwise ranking-oriented loss functions from LTR. From the perspective of LTR, the CF methods with pointwise losses and the CR methods with pairwise losses can be referred to as pointwise and pairwise approaches, respectively.

CIKM'17, November 6–10, 2017, Singapore.
 ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: http://dx.doi.org/10.1145/3132847.3133100

Despite the successes of the existing pointwise and pairwise methods, the relations and differences between them are rarely investigated in the literature. Basically, both types of methods adopt the same MF scoring function to describe the relevance between a user and an item. The major difference is how to learn the scoring function: by minimizing either a pointwise loss or a pairwise loss. The pointwise methods focus on learning a user's absolute interest on a single item (fitting the rating), while the pairwise methods emphasize to learn his/her relative preference on a pair of items (fitting the order). The distinct fitting traits of them determine their different properties and strengths embodied in two specific aspects. First, the trained scoring functions are searched stochastically from two distinct solution spaces, with different probabilities of dissatisfaction. Second, the collaboration patterns between any two users, determining how similar they will be, are different during the learning process. In summary, it will be much beneficial if these different properties can be integrated so as to achieve accurate rating and order simultaneously.

In this paper, we develop a novel joint loss function to model both absolute interests and relative preferences of users. The proposed joint loss is defined over pairwise training examples, which consists of four components: two pointwise losses, a pairwise loss and a regularization term. For optimization, we develop a novel alternating learning algorithm with stochastic gradient descent (SGD), where pointwise and pairwise SGD updates are conducted, alternately and interactively, based on a uniformly drawn pairwise example. We name the proposed joint learning method alternating pointwise-pairwise learning (APPL). APPL utilizes two distinct fitting traits simultaneously and effectively, thus is able to force the learning algorithm to search a better scoring function from a smaller solution subspace, as well as to integrate two different collaboration patterns. We compare APPL with a variety of existing pointwise and pairwise methods, as well as two joint methods, on four real-world datasets. The extensive experiments with both explicit and implicit feedback settings show that it performs significantly better than the competitors in most cases.

2 THE PROPOSED METHOD

2.1 Preliminaries

According to the type of user feedback, explicit or implicit, personalized item ranking can be divided into two distinct problems: explicit feedback ranking and implicit feedback ranking. In the explicit setting, the observed feedback of a user on an item is expressed by a graded rating, indicating his/her explicit preference, and the unobserved case is usually omitted in the model. While in the implicit setting, only positive feedback can be observed, denoting by 1, and the unobserved case is usually assumed as negative feedback, denoting by 0.

In this paper, we use the explicit feedback setting as an example to describe our proposed method. For implicit setting, the notations and meanings will be similar. We adopt a typical 5-star rating system with rating 1, 2, 3, 4 and 5, where 1 and 5 denotes the lowest and highest relevance between a user and an item, respectively. Suppose we have a recommender system which involves m users and n items. Let $R \in \mathbb{R}^{m \times n}$ denotes the observed user-item matrix. Let R_{ui}, where $R_{ui} \neq 0$, denotes a valid pointwise training example, indicating the rating of user u on item i. Let (R_{ui}, R_{uj}), where $R_{ui} > R_{uj}$, denotes a valid pairwise training example, indicating user u prefers item i to item j.

2.2 Pointwise and Pairwise Learning

MF scoring functions are widely-used in existing pointwise and pairwise methods [1, 6, 7, 14, 15], to model/predict the relevance between a user and an item. A standard MF scoring function is defined as:

$$\hat{R}_{ui} = U_u^{\mathrm{T}} V_i, \tag{1}$$

where the d-dimensional vectors U_u and V_i denote the latent feature vectors of user u and item i, respectively.

To learn the MF scoring function $\hat{R}_{ui}$, a pointwise squared loss with L_2 regularization derived from probabilistic matrix factorization (PMF) [15], is defined over the pointwise example R_{ui}:

$$\mathcal{L}_{point}(R_{ui}) = (\hat{R}_{ui} - R_{ui})^2 + \lambda(\|U_u\|_2^2 + \|V_i\|_2^2), \tag{2}$$

where λ denotes the regularization parameter.

In contrast, a pairwise logarithmic loss with L_2 regularization derived from Bayesian personalized ranking (BPR) [14], is defined over the pairwise example (R_{ui}, R_{uj}):

$$\mathcal{L}_{pair}(R_{ui}, R_{uj}) = \Delta R_{ui,uj} \cdot \log(1 + \exp(-\Delta \hat{R}_{ui,uj})) + \lambda(\|U_u\|_2^2 + \|V_i\|_2^2 + \|V_j\|_2^2), \tag{3}$$

where $\Delta R_{ui,uj} = R_{ui} - R_{uj}$ and $\Delta \hat{R}_{ui,uj} = \hat{R}_{ui} - \hat{R}_{uj}$.

2.3 Insights into the Pointwise and Pairwise Learning Methods

Intuitively, the goal of both pointwise and pairwise learning is to search a good MF scoring function, i.e., a good trained model. Minimizing a pointwise loss such as $\mathcal{L}_{point}$ is going to learn a model that well fits the absolute rating R_{ui}. While minimizing a pairwise loss such as $\mathcal{L}_{pair}$ is equivalent to maximizing $\Delta \hat{R}_{ui,uj}$, which is going to learn a model that well fits the relative order of R_{ui} and R_{uj}, such that item i will be ranked above item j for user u. Due to the distinct fitting traits, the differences between pointwise and pairwise methods are mainly embodied in two aspects: the model learning for each single user, and the collaboration pattern between any two users.

2.3.1 Differences in Model Learning. The loss is usually minimized to a specific level with arbitrary non-zero errors, due to the complicated properties of real-world problems. As a result, either a pointwise or pairwise method will produce a solution space of possible trained models, but only a small part of them are good ones. The final trained model is stochastically searched from the solution space, which means it may be undesired with a certain probability. Obviously, the solution spaces of pointwise and pairwise methods are different.

As shown in Figure 1, suppose we have a user and two items A and B with the ground-truth ratings 4 and 3, respectively. If $\mathcal{L}_{point}$ can be minimized to a level that the absolute error $|\hat{R}_{ui} - R_{ui}| = 0.7$ for all pointwise examples, the pointwise solution space will be {M1, M2, M3, M4} (all fulfill the training error). However, only M3 and M4 are desired trained models since they can predict both the correct ranking and the consistent direction with the ground-truth, i.e., $sign(\hat{R}_{ui} - R_{ui}) = sign(\hat{R}_{uj} - R_{uj})$. The latter condition is critical when considering more than two items. For example, if the model M2 predict 2.7 for an additional item C with ground-truth rating 2, it will provide an incorrect ranking of B and C.

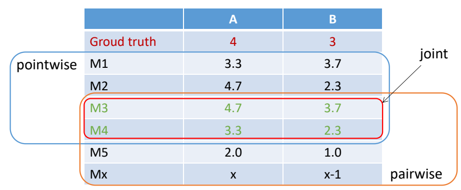

Figure 1: The solution spaces of different methods.

Similarly, if $\mathcal{L}_{pair}$ can be minimized to the same level such that $\Delta \hat{R}_{ui,uj} = 1.0$ for all pairwise examples, the pairwise solution space will be {M3, M4, M5,...,Mx}. Again, only M3 and M4 are desired trained models, as M5,...,Mx cannot provide the good predictions with small absolute errors against the ground-truth. Without the constraint of small absolute errors, the trained model will be any one among M5,..., Mx for an item pair. For example, the model may be M5 for item pair (A, B), and Mx for another pair (C, D). This will disorder the ranking of pair (A, C), (A, D), or (B, C), etc. All item pairs have to be compared in order to train a unified model that ranks correctly for all items. This is inefficient or even impossible when the item space becomes large.

In summary, the different solution spaces of pointwise and pairwise methods have an overlapped subspace in which most of trained models are desired. Thus, it is desirable to develop a joint method that can reduce the solution space to the intersection part.

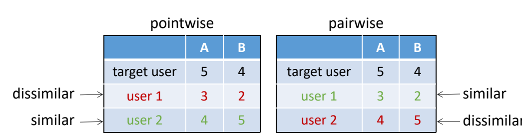

Figure 2: The collaboration patterns of different methods.

2.3.2 Differences in Collaboration Pattern. For both pointwise and pairwise methods, by means of the MF scoring function, all users collaborate/interact with each other in the same vector space during the learning process. However, the collaboration patterns of them are different, based on a single common-rated item and a pair of common-rated items, respectively. This difference also affects the final trained model, since it determines how similar or dissimilar the trained feature vectors of the two users will be. As shown in Figure 2, suppose we have a rating matrix with three users and two items. For pointwise methods, the trained feature vectors of user 2 and target user will be much more similar, compared to that of user 1 and target user. This is because user 2 has relatively close ground-truth ratings with the target user. For pairwise methods, in contrast, user 1 and target user will be more similar, since they both prefer item A to item B. Thus, it will be beneficial if the impacts of the different collaboration patterns can be well balanced.

2.4 A Novel Joint Learning Method

Motivated by above analysis, we attempt to develop a joint learning method that can take advantage of both pointwise and pairwise learning. However, how to achieve benefits from the combination, especially how to reduce the solution space, are big challenges. An intuitive idea is probably to fuse (e.g., average) the predicted scores of the two models trained independently by $\mathcal{L}_{point}$ and $\mathcal{L}_{pair}$. Though this fusion method obtains benefits by fusing the results, it fails to reduce the solution space during the learning process, and cannot be deemed as a joint learning method.

Instead, we develop a novel joint loss function to perform joint learning. More specifically, a joint loss over the pairwise example (R_{ui}, R_{uj}) is defined as:

$$\begin{aligned}\mathcal{L}_{joint}(R_{ui}, R_{uj}) = (\hat{R}_{ui} - R_{ui})^2 + (\hat{R}_{uj} - R_{uj})^2 \\ + \Delta R_{ui,uj} \cdot \log(1 + \exp(-\Delta \hat{R}_{ui,uj})) \\ + \lambda(\|U_u\|_2^2 + \|V_i\|_2^2 + \|V_j\|_2^2). \qquad (4)\end{aligned}$$

The proposed joint loss $\mathcal{L}_{joint}$ consists of four components: two pointwise PMF losses, a pairwise BPR loss, and a L_2 regularization term. It describes both the absolute rating information of R_{ui} and R_{uj}, and the relative order information of them.

To optimize $\mathcal{L}_{joint}$, a common practice is to employ stochastic gradient descent (SGD) with bootstrapping [14]. However, directly optimizing $\mathcal{L}_{joint}$ over pairwise examples may not be efficient due to its heterogeneity. Thus, we propose a novel alternating learning algorithm to minimize the pointwise loss $\mathcal{L}_{point}$ and the pairwise loss $\mathcal{L}_{pair}$, alternately. More specifically, in each iteration, for a uniformly drawn pairwise example (R_{ui}, R_{uj}), we first perform two pointwise SGD updates based on $\mathcal{L}_{point}(R_{ui})$ and $\mathcal{L}_{point}(R_{uj})$, then perform a pairwise SGD update based on $\mathcal{L}_{pair}(R_{ui}, R_{uj})$. We name this novel joint learning method alternating pointwise-pairwise learning (APPL). The pseudocode of the APPL learning algorithm is presented in Algorithm 1.

Algorithm 1 The APPL Learning Algorithm

Input: data R, dimensionality d, regularization λ, learning rate α
1. Initialize U, V with Gaussian distribution $N(0, 0.01)$
2. **while** not converged **do**
3. Uniformly draw a pairwise training example (R_{ui}, R_{uj})
4. Update U_u, V_i by SGD based on $\mathcal{L}_{point}(R_{ui})$
5. Update U_u, V_j by SGD based on $\mathcal{L}_{point}(R_{uj})$
6. Update U_u, V_i, V_j by SGD based on $\mathcal{L}_{pair}(R_{ui}, R_{uj})$
7. **end while**

Output: the trained feature vectors U, V

During the learning process of APPL, the feature vectors are updated by pointwise and pairwise losses, alternately and interactively. It utilizes two distinct fitting traits, and is able to force the learning algorithm to search a better trained model from a smaller solution space, as well as to integrate two different collaboration patterns. Furthermore, APPL can be deemed as a general joint learning framework in which more complex techniques can be incorporated. For example, the pairwise loss $\mathcal{L}_{pair}$ can be replaced by other losses such as hinge and exponential losses [7]. Also, the techniques such as group preference [11], local low-rank approximation [8], multi-channel feedback [10] and social relationships [12], can be combined.

3 EXPERIMENTS AND VALIDATIONS

In this section, we validate the proposed joint method APPL, and mainly focus on whether it can achieve better performance than pointwise, pairwise or joint methods, for either the explicit or implicit feedback setting.

3.1 Experimental Setup

3.1.1 Datasets. We employ four real-world datasets for experiments, including two MovieLens datasets[1]: ML100K and ML1M, and two Yahoo! datasets[2]: Yahoo!R3 and Yahoo!R4. All the datasets contain explicit ratings of users to items. For the Yahoo! datasets, we remove the users who have fewer than 20 ratings in order to be consistent with the MovieLens datasets, as well as to ensure their are enough testing ratings for the explicit feedback setting. For the implicit setting, all the observed ratings are converted to 1. In summary, the statistics of all datasets are presented in Table 1.

Table 1: The statistics of datasets.

Statistics	ML100K	ML1M	Yahoo!R3	Yahoo!R4
#users	943	6,040	5,050	2,867
#items	1,682	3,952	1,000	10,825
#ratings	100,000	1,000,209	174,497	145,034
density	6.30%	4.19%	3.46%	0.47%

3.1.2 Evaluation Methodology. We employ cross validation for the training and testing. For each user, 50% ratings are randomly selected for the training and the remaining 50% for the testing. For each dataset, we repeat the above process 5 times independently and obtain 5 splits of data. The experimental results are averaged over these 5 splits. We adopt two widely-used ranking metrics for the evaluation: the normalized discounted cumulative gain NDCG@N ($N = 5, 10$) for the explicit feedback setting, and the mean average precision MAP@N ($N = 10, 20$) for the implicit feedback setting.

3.1.3 Baselines for Comparison. We compare APPL with a number of state-of-the-art methods, including BPMF [16], BPR [14], GAPFM [17], GCR [7], LLORMA [8], PMF [15] and SVD++ [5] for explicit feedback ranking, and AOBPR [13], BPR, FISMAUC [4], GBPR [11] and PMF for implicit feedback ranking. We use the implementation versions in the public library LibRec [3], except the cases of BPR, GAPFM and GCR for the explicit setting, and PMF for the implicit setting. Moreover, we also compare APPL with two joint methods: the score fusion method (named Fusion) and the direct optimization method (named Direct), as discussed in Section 2.4.

[1]http://grouplens.org/datasets/movielens/
[2]http://webscope.sandbox.yahoo.com/catalog.php?datatype=r

Since all methods adopt the MF scoring function, we set the dimensionality of feature vector $d = 10$. For other model parameters, the optimal values are tuned based on the cross validation.

3.2 Experimental Results

3.2.1 Results of Explicit Feedback Ranking. We first compare the performance of all methods for explicit feedback ranking. The NDCG@N ($N = 5, 10$) results on all datasets are presented in Table 2, where the bold font indicates the best performance in each case. The proposed APPL works best in all cases. The conducted paired t-test demonstrates that its advantages over most of the competitors, including the joint methods Fusion and Direct, are statistically considerable, at the significance level of 0.05 ($p < 0.05$). In particular, it performs much better than the pointwise method PMF and the pairwise method BPR. For example, in terms of NDCG@5, it outperforms PMF by 4.08%, 1.81%, 5.17% and 3.28%, and outperforms BPR by 2.03%, 2.90%, 6.85% and 2.22%, on the datasets ML100K, ML1M, Yahoo!R3 and Yahoo!R4, respectively.

Table 2: Comparison for explicit feedback ranking.

	Types	Methods	ML100K	ML1M	Yahoo!R3	Yahoo!R4
NDCG@5	point	BPMF	0.7272	0.7850	0.6512	0.8551
		LLORMA	0.7219	0.7688	0.6571	0.8469
		PMF	0.7003	0.7742	0.6366	0.8334
		SVD++	0.7214	0.7771	0.6612	0.8454
	pair	BPR	0.7144	0.7660	0.6266	0.8420
		GAPFM	0.7042	0.7465	0.6381	0.8510
		GCR	0.7161	0.7783	0.6248	0.8470
	joint	Fusion	0.7250	0.7847	0.6502	0.8511
		Direct	0.7258	0.7826	0.6596	0.8551
		APPL	**0.7289**	**0.7882**	**0.6695**	**0.8607**
NDCG@10	point	BPMF	0.7596	0.7948	0.7466	0.8816
		LLORMA	0.7566	0.7822	0.7517	0.8747
		PMF	0.7401	0.7864	0.7350	0.8646
		SVD++	0.7553	0.7880	0.7550	0.8727
	pair	BPR	0.7493	0.7797	0.7287	0.8718
		GAPFM	0.7407	0.7607	0.7346	0.8737
		GCR	0.7532	0.7895	0.7277	0.8753
	joint	Fusion	0.7579	0.7932	0.7447	0.8770
		Direct	0.7572	0.7923	0.7521	0.8797
		APPL	**0.7610**	**0.7970**	**0.7603**	**0.8838**

3.2.2 Results of Implicit Feedback Ranking. We now compare the performance of all methods for implicit feedback ranking. The MAP@N ($N = 10, 20$) results are shown in Table 3. For this different task, the proposed APPL still works much better than the competitor methods in most cases. For example, in terms of MAP@10, it surpasses PMF by 62.74%, 63.76% and 55.00%, and surpasses BPR by 7.66%, 3.22% and 17.20%, on the datasets ML100K, ML1M and Yahoo!R3, respectively. While on the Yahoo!R4 dataset, the score fusion method Fusion performs best. The degradation of APPL and Direct in this case is probably due to the low density of positive feedback data, where the joint learning seems hard to accomplish.

4 CONCLUSIONS

In this paper, we analyze the different properties of pointwise and pairwise personalized ranking methods. We develop a novel joint learning method named APPL, which takes advantage of both pointwise and pairwise learning. We conduct extensive experiments to validate APPL by the comparison with a large number of existing methods on four real-world datasets. The results of both explicit and implicit feedback ranking demonstrate the effectiveness of APPL.

Table 3: Comparison for implicit feedback ranking.

	Types	Methods	ML100K	ML1M	Yahoo!R3	Yahoo!R4
MAP@10	point	PMF	0.2496	0.2307	0.1231	0.1558
	pair	AOBPR	0.3620	0.3627	0.1621	0.1714
		BPR	0.3773	0.3660	0.1628	0.1773
		FISMAUC	0.2435	0.2190	0.1220	0.1423
		GBPR	0.3841	0.3535	0.1788	0.1637
	joint	Fusion	0.4001	0.3323	0.1859	**0.1904**
		Direct	0.3745	0.3432	0.1759	0.1466
		APPL	**0.4062**	**0.3778**	**0.1908**	0.1751
MAP@20	point	PMF	0.2011	0.1880	0.1082	0.1302
	pair	AOBPR	0.3145	0.3047	0.1435	0.1561
		BPR	0.3276	0.3075	0.1440	0.1603
		FISMAUC	0.1996	0.1819	0.1073	0.1130
		GBPR	0.3315	0.2939	0.1578	0.1400
	joint	Fusion	0.3413	0.2623	0.1617	**0.1665**
		Direct	0.3256	0.2910	0.1555	0.1362
		APPL	**0.3501**	**0.3106**	**0.1680**	0.1526

ACKNOWLEDGMENTS

The work was supported by Research Grants Council of Hong Kong (PolyU 5202/12E, PolyU 152094/14E), National Natural Science Foundation of China (61272291) and The Hong Kong Polytechnic University (4-BCB5, G-YBJP).

REFERENCES

[1] Suhrid Balakrishnan and Sumit Chopra. 2012. Collaborative ranking. In *WSDM*. ACM, 143–152.
[2] Konstantina Christakopoulou and Arindam Banerjee. 2015. Collaborative ranking with a push at the top. In *WWW*. ACM, 205–215.
[3] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. 2015. LibRec: A Java Library for Recommender Systems.. In *UMAP Workshops*.
[4] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *SIGKDD*. ACM, 659–667.
[5] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*. ACM, 426–434.
[6] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.
[7] Joonseok Lee, Samy Bengio, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2014. Local collaborative ranking. In *WWW*. ACM, 85–96.
[8] Joonseok Lee, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2013. Local low-rank matrix approximation. In *ICML*. 82–90.
[9] Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3, 3 (2009), 225–331.
[10] Babak Loni, Roberto Pagano, Martha Larson, and Alan Hanjalic. 2016. Bayesian Personalized Ranking with Multi-Channel User Feedback. In *RecSys*. ACM, 361–364.
[11] Weike Pan and Li Chen. 2013. GBPR: Group Preference Based Bayesian Personalized Ranking for One-Class Collaborative Filtering. In *IJCAI*, Vol. 13. 2691–2697.
[12] Dimitrios Rafailidis and Fabio Crestani. 2016. Joint Collaborative Ranking with Social Relationships in Top-N Recommendation. In *CIKM*. ACM, 1393–1402.
[13] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *WSDM*. ACM, 273–282.
[14] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. AUAI Press, 452–461.
[15] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic matrix factorization. In *NIPS*. 1257–1264.
[16] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*. ACM, 880–887.
[17] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, and Alan Hanjalic. 2013. GAPfm: Optimal top-n recommendations for graded relevance domains. In *CIKM*. ACM, 2261–2266.