# CoFiSet: Collaborative Filtering via Learning Pairwise Preferences over Item-sets

Weike Pan* Li Chen*

**Abstract**

Collaborative filtering aims to make use of users' feedbacks to improve the recommendation performance, which has been deployed in various industry recommender systems. Some recent works have switched from exploiting explicit feedbacks of numerical ratings to implicit feedbacks like browsing and shopping records, since such data are more abundant and easier to collect. One fundamental challenge of leveraging implicit feedbacks is the lack of negative feedbacks, because there are only some observed relatively "positive" feedbacks, making it difficult to learn a prediction model. Previous works address this challenge via proposing some pointwise or pairwise preference assumptions on items. However, such assumptions with respect to items may not always hold, for example, a user may dislike a bought item or like an item not bought yet. In this paper, we propose a new and relaxed assumption of *pairwise preferences over item-sets*, which defines a user's preference on a set of items (item-set) instead of on a single item. The relaxed assumption can give us more accurate pairwise preference relationships. With this assumption, we further develop a general algorithm called CoFiSet (**co**llaborative **fi**ltering via learning pairwise preferences over item-**set**s). Experimental results show that CoFiSet performs better than several state-of-the-art methods on various ranking-oriented evaluation metrics on two real-world data sets. Furthermore, CoFiSet is very efficient as shown by both the time complexity and CPU time.

**Keywords:** Pairwise Preferences over Item-sets, Collaborative Filtering, Implicit Feedbacks

## 1 Introduction

Collaborative filtering [4] as a content free technique has been widely adopted in commercial recommender systems [2, 11]. Various model-based methods have been proposed to improve the prediction accuracy using users' explicit feedbacks such as numerical ratings [9, 13, 16] or transferring knowledge from auxiliary data [10, 15]. However, in real applications, users' explicit ratings are not easily obtained, so they are not sufficient for the purpose of training an adequate prediction model, while users' implicit data like browsing and shopping records can be more easily collected. Some recent works have thus turned to improve the recommendation performance via exploiting users' implicit feedbacks, which include users' logs of clicking social updates [5], watching TV programs [6], assigning tags [14], purchasing products [17], browsing web pages [20], etc.

One fundamental challenge in collaborative filtering with implicit feedbacks is the lack of negative feedbacks. A learning algorithm can only make use of some observed relatively "positive" feedbacks, instead of ordinal ratings in explicit data. Some early works [6, 14] assume that an observed feedback denotes "like" and an unobserved feedback denotes "dislike", and propose to reduce the problem to collaborative filtering with explicit feedbacks via some weighting strategies. Recently, some works [17, 19] assume that a user prefers an observed item to an unobserved item, and reduce the problem to a classification [17] or a regression [19] problem. Empirically, the latter assumption of pairwise preferences over two items results in better recommendation accuracy than the earlier like/dislike assumption.

However, the pairwise preferences with respect to two items might not be always valid. For example, a user bought some fruit but afterwards he finds that he actually does not like it very much, or a user may inherently like some fruit though he has not bought it yet. In this paper, we propose a new and relaxed assumption, which is that *a user is likely to prefer a set of observed items to a set of unobserved items*. We call our assumption *pairwise preferences over item-sets*, which is illustrated in Figure 1. In Figure 1, we can see that the pairwise preference relationship of "apple $\succ$ peach" does not hold for this user, since his true preference score on apple is lower than that on peach. On the contrary, the relaxed pairwise relationship of "item-set of apple and grapes $\succ$ item-set of peach" is more likely to be true, since he likes grapes a lot. Thus, we can see that our assumption is more accurate and the corresponding pairwise relationship is more likely to be valid. With this assumption, we define a user's

*Department of Computer Science, Hong Kong Baptist University, Hong Kong. {wkpan,lichen}@comp.hkbu.edu.hk

preference to be on a set of items (item-set) rather than on a single item, and then develop a general algorithm called CoFiSet. Note that we use the term "item-set" instead of "itemset" to make it different from that in frequent pattern mining [8].

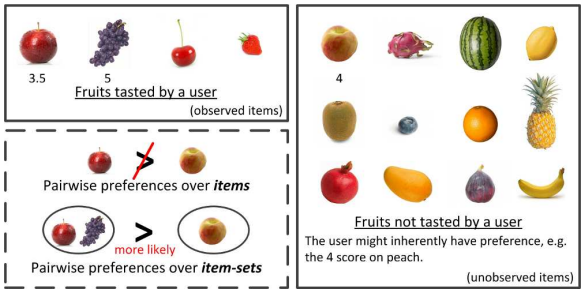



Figure 1: Illustration of *pairwise preferences over item-sets*. The numbers under some fruit denote a user's true preference scores, $\hat{r}_{u,apple} = 3.5$, $\hat{r}_{u,grapes} = 5$ and $\hat{r}_{u,peach} = 4$. We thus have the relationships $\hat{r}_{u,apple} \not> \hat{r}_{u,peach}$ and $(\hat{r}_{u,apple} + \hat{r}_{u,grapes})/2 > \hat{r}_{u,peach}$.

We summarize our main contributions as follows, (1) we define a user's preference on an item-set (a set of items) instead of on a single item, since there is likely high uncertainty of a user's item-level preference in implicit data; (2) we propose a new and relaxed assumption, *pairwise preferences over item-sets*, to fully exploit users' implicit data; (3) we develop a general algorithm, CoFiSet, which absorbs some recent algorithms as special cases; and (4) we conduct extensive empirical studies, and observe better recommendation performance of CoFiSet than several state-of-the-art methods [17, 19, 20].

## 2 Learning Pairwise Preferences over Item-sets

**2.1 Problem Definition** Suppose we have some observed feedbacks, $\mathcal{R}^{tr} = \{(u, i)\}$, from $n$ users and $m$ items. Our goal is then to recommend a personalized ranking list of items for each user $u$. Our studied problem is usually called one-class collaborative filtering [14] or collaborative filtering with implicit feedbacks [6, 17] in general. We list some notations used in the paper in Table 1.

**2.2 Preference Assumption** Collaborative filtering with implicit feedbacks is quite different from the task of 5-star numerical rating estimation [9], since there are only some observed relatively "positive" feedbacks, making it difficult to learn a prediction model [6, 14, 17]. So far, there have been mainly two types of assumptions proposed to model the implicit feedbacks, *pointwise preference on an item* [6, 14], and *pairwise preferences over two items* [17]. We first describe these two types of assumptions formally, and then propose a new and relaxed assumption.

Table 1: Some notations used in the paper.

| Notation | Description |
|---|---|
| $\mathcal{U}^{tr} = \{u\}_{u=1}^{n}$ | training user set |
| $\mathcal{U}_i^{tr}$ | training user set w.r.t. item $i$ |
| $\mathcal{U}^{te} \subseteq \mathcal{U}^{tr}$ | test user set |
| $\mathcal{I}^{tr} = \{i\}_{i=1}^{m}$ | training item set |
| $\mathcal{I}_u^{tr}$ | training item set w.r.t. user $u$ |
| $\mathcal{I}_u^{te}$ | test item set w.r.t. user $u$ |
| $\mathcal{P} \subseteq \mathcal{I}^{tr}$ | item set (*presence* of observation) |
| $\mathcal{A} \subseteq \mathcal{I}^{tr}$ | item set (*absence* of observation) |
| $u \in \mathcal{U}^{tr}$ | user index |
| $i, i', j \in \mathcal{I}^{tr}$ | item index |
| $\mathcal{R}^{tr} = \{(u, i)\}$ | training data |
| $\mathcal{R}^{te} = \{(u, i)\}$ | test data |
| $\hat{r}_{ui}$ | preference of user $u$ on item $i$ |
| $\hat{r}_{u\mathcal{P}}$ | preference of user $u$ on item-set $\mathcal{P}$ |
| $\hat{r}_{u\mathcal{A}}$ | preference of user $u$ on item-set $\mathcal{A}$ |
| $\hat{r}_{uij}, \hat{r}_{ui\mathcal{A}}, \hat{r}_{u\mathcal{P}\mathcal{A}}$ | pairwise preferences of user $u$ |
| $\Theta$ | set of model parameters |
| $U_{u\cdot} \in \mathbb{R}^{1\times d}$ | user $u$'s latent feature vector |
| $V_{i\cdot} \in \mathbb{R}^{1\times d}$ | item $i$'s latent feature vector |
| $b_i \in \mathbb{R}$ | item $i$'s bias |

The assumption of pointwise preference on an item [6, 14] can be represented as follows,

$$\hat{r}_{ui} = 1, \hat{r}_{uj} = 0,\ i \in \mathcal{I}_u^{tr}, j \in \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr}, \tag{2.1}$$

where 1 and 0 are used to denote "like" and "dislike" for an observed (user, item) pair and an unobserved (user, item) pair, respectively. With this assumption, confidence-based weighting strategies are incorporated into the objective function [6, 14]. However, finding a good weighting strategy for each observed feedback is still a very difficult task in real applications. Furthermore, treating all observed feedbacks as "likes" and unobserved feedbacks as "dislikes" may mislead the learning process.

The assumption of pairwise preferences over two items [17] relax the assumption of pointwise preferences [6, 14], which can be represented as follows,

$$\hat{r}_{ui} > \hat{r}_{uj},\ i \in \mathcal{I}_u^{tr}, j \in \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr} \tag{2.2}$$

where the relationship $\hat{r}_{ui} > \hat{r}_{uj}$ means that a user $u$ is likely to prefer an item $i \in \mathcal{I}_u^{tr}$ to an item $j \in$

$\mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}$. Empirically this assumption generates better recommendation results than that of [6, 14].

However, as mentioned in the introduction, in real situations, such pairwise assumption may not hold for each item pair $(i,j), i \in \mathcal{I}_u^{tr}, j \in \mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}$. Specifically, there are two phenomena: first, there may exist some item $i \in \mathcal{I}_u^{tr}$ that user $u$ does not like very much; second, there may exist some item $j \in \mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}$ that user $u$ likes but has not found yet, which also motivates a recommender system to help user explore the items. The second case is more likely to occur since a user's preferences on items from $\mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}$ are usually not the same, including both "likes" and "dislikes". In either of the above two cases, the relationship $\hat{r}_{ui} > \hat{r}_{uj}$ in Eq.(2.2) does not hold. Thus, the assumption of *pairwise preferences over items* [17] may not be true for all of item pairs.

Before we present a new type of assumption, we first introduce two definitions, a user $u$'s preference on an item-set and pairwise preferences over two item-sets.

DEFINITION 2.1. A user $u$'s preference on an item-set (a set of items) is defined as a function of user $u$'s preferences on items in the item-set. For example, user $u$'s preference on item-set $\mathcal{P}$ can be $\hat{r}_{u\mathcal{P}} = \sum_{i\in\mathcal{P}} \hat{r}_{ui}/|\mathcal{P}|$, or in other forms.

DEFINITION 2.2. A user $u$'s pairwise preferences over two item-sets is defined as the difference of user $u$'s preferences on two item-sets. For example, user $u$'s pairwise preferences over item-sets $\mathcal{P}$ and $\mathcal{A}$ can be $\hat{r}_{u\mathcal{P}\mathcal{A}} = \hat{r}_{u\mathcal{P}} - \hat{r}_{u\mathcal{A}}$, or in other forms.

With the above two definitions, we further relax the assumption of *pairwise preferences over items* made in [17] and propose a new one called *pairwise preferences over item-sets*,

$$\hat{r}_{u\mathcal{P}} > \hat{r}_{u\mathcal{A}},\ \mathcal{P} \subseteq \mathcal{I}_u^{tr}, \mathcal{A} \subseteq \mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr} \tag{2.3}$$

where $\hat{r}_{u\mathcal{P}}$ and $\hat{r}_{u\mathcal{A}}$ are the user $u$'s overall preferences on the items from item-set $\mathcal{P}$ and item-set $\mathcal{A}$, respectively. For a user $u$, $\mathcal{P} \subseteq \mathcal{I}_u^{tr}$ denotes a set of items with observed feedbacks from user $u$ (*presence* of observation), and $\mathcal{A} \subseteq \mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}$ denotes a set of items without observed feedbacks from user $u$ (*absence* of observation). In our assumption, the granularity of pairwise preference is the item-set instead of the item, which should be closer to real situations. Our proposed assumption is also more general and can embody the assumption of *pairwise preferences over items* [17] as a special case.

**2.3 Model Formulation** Assuming that a user $u$ is likely to prefer an item-set $\mathcal{P}$ to an item-set $\mathcal{A}$, we may introduce a constraint $\hat{r}_{u\mathcal{P}} > \hat{r}_{u\mathcal{A}}$ when learning the parameters of the prediction model. Specifically, for a pair of item-sets $\mathcal{P}$ and $\mathcal{A}$, we have the following optimization problem,

$$\min_{\Theta}\ \mathcal{R}(u,\mathcal{P},\mathcal{A}),\ \text{s.t. } \hat{r}_{u\mathcal{P}} > \hat{r}_{u\mathcal{A}}$$

where the hard constraint $\hat{r}_{u\mathcal{P}} > \hat{r}_{u\mathcal{A}}$ is based on a user's pairwise preferences over item-sets, and $\mathcal{R}(u,\mathcal{P},\mathcal{A})$ is a regularization term used to avoid overfitting. Since the above optimization problem is difficult to solve due to the hard constraint, we relax the constraint, and introduce a loss term in the objective function,

$$\min_{\Theta}\ \mathcal{L}(u,\mathcal{P},\mathcal{A}) + \mathcal{R}(u,\mathcal{P},\mathcal{A}),$$

where $\mathcal{L}(u,\mathcal{P},\mathcal{A})$ is the loss term w.r.t. user $u$'s preferences on item-sets $\mathcal{P}$ and $\mathcal{A}$. Then, for each user $u$, we have the following optimization problem,

$$\min_{\Theta} \sum_{\mathcal{P}\subseteq\mathcal{I}_u^{tr}} \sum_{\mathcal{A}\subseteq\mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}} \mathcal{L}(u,\mathcal{P},\mathcal{A}) + \mathcal{R}(u,\mathcal{P},\mathcal{A}),$$

where $\mathcal{P}$ is a subset of items randomly sampled from $\mathcal{I}_u^{tr}$ that denotes a set of items with observed feedbacks from user $u$, and $\mathcal{A}$ is a subset of items randomly sampled from $\mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}$ that denotes a set of items without observed feedbacks from user $u$.

Finally, to encourage collaborations among the users, we reach the following optimization problem for all users in training data $\mathcal{R}^{tr} = \{(u,i)\}$,

$$\min_{\Theta} \sum_{u\in\mathcal{U}^{tr}} \sum_{\mathcal{P}\subseteq\mathcal{I}_u^{tr}} \sum_{\mathcal{A}\subseteq\mathcal{I}^{tr}\backslash\mathcal{I}_u^{tr}} \mathcal{L}(u,\mathcal{P},\mathcal{A}) + \mathcal{R}(u,\mathcal{P},\mathcal{A}), \tag{2.4}$$

where $\Theta = \{U_{u\cdot}, V_{i\cdot}, b_i, u \in \mathcal{U}^{tr}, i \in \mathcal{I}^{tr}\}$ denotes the parameters to be learned. The loss term $\mathcal{L}(u,\mathcal{P},\mathcal{A})$ is defined on the user $u$'s pairwise preferences over item-sets, $\hat{r}_{u\mathcal{P}\mathcal{A}} = \hat{r}_{u\mathcal{P}} - \hat{r}_{u\mathcal{A}}$, where $\hat{r}_{u\mathcal{P}} = \sum_{i\in\mathcal{P}} \hat{r}_{ui}/|\mathcal{P}|$ and $\hat{r}_{u\mathcal{A}} = \sum_{j\in\mathcal{A}} \hat{r}_{uj}/|\mathcal{A}|$. The regularization term $\mathcal{R}(u,\mathcal{L},\mathcal{A}) = \frac{\alpha_u}{2}\|U_{u\cdot}\|^2 + \sum_{i\in\mathcal{P}}[\frac{\alpha_v}{2}\|V_{i\cdot}\|^2 + \frac{\beta_v}{2}\|b_i\|^2] + \sum_{j\in\mathcal{A}}[\frac{\alpha_v}{2}\|V_{j\cdot}\|^2 + \frac{\beta_v}{2}\|b_j\|^2]$ is used to avoid overfitting during parameter learning, and $\alpha_u$, $\alpha_v$, $\beta_v$ are hyper-parameters.

Note again that the core concept in our preference assumption and objective function is "item-set" (a set of items), not "item" in [6, 14, 17]. For this reason, we call our solution as CoFiSet (**co**llaborative **fi**ltering via learning pairwise preferences over item-**set**s). Another notice is that the loss term in CCF(Hinge) [20] can be equivalently written as pairwise preferences over an item $i$ and an item-set $\mathcal{A}$, $\hat{r}_{ui\mathcal{A}}$, which is a special case of our CoFiSet. CCF(SoftMax) [20] can only be written

as pairwise preferences, $\hat{r}_{uij}$, over items $i$ and $j \in \mathcal{A}$. In both CCF(Hinge) [20] and CCF(SoftMax) [20], item $i$ is considered as a preferred or chosen one given a candidate set $i \cup \mathcal{A}$, which is motivated from industry recommender systems with impression data as users' choice context.

**2.4 Learning the CoFiSet** We adopt the widely used SGD (stochastic gradient descent) algorithmic framework in collaborative filtering [9] to learn the model parameters. We first derive the gradients and update rules for each variable.

We have the gradients of the variables w.r.t. the loss term $\mathcal{L}(u, \mathcal{P}, \mathcal{A})$: $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial U_{u\cdot}} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}}(\bar{V}_{\mathcal{P}\cdot} - \bar{V}_{\mathcal{A}\cdot})$; $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial V_{i\cdot}} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}} \frac{U_{u\cdot}}{|\mathcal{P}|}$, $i \in \mathcal{P}$; $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial V_{j\cdot}} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}} \frac{-U_{u\cdot}}{|\mathcal{A}|}$, $j \in \mathcal{A}$; $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial b_i} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}} \frac{1}{|\mathcal{P}|}$, $i \in \mathcal{P}$; and $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial b_j} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}} \frac{-1}{|\mathcal{A}|}$, $j \in \mathcal{A}$, where $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}} = -\sigma(-\hat{r}_{u\mathcal{P}\mathcal{A}})$. $\bar{V}_{\mathcal{P}\cdot} = \sum_{i\in\mathcal{P}} V_{j\cdot}/|\mathcal{P}|$ and $\bar{V}_{\mathcal{A}\cdot} = \sum_{j\in\mathcal{A}} V_{j\cdot}/|\mathcal{A}|$ are the average latent feature representation of items in $\mathcal{P}$ and $\mathcal{A}$, respectively.

We have the gradients of the variables w.r.t. the regularization term $\mathcal{R}(u, \mathcal{P}, \mathcal{A})$: $\frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial U_{u\cdot}} = \alpha_u U_{u\cdot}$; $\frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial V_{i\cdot}} = \alpha_v V_{i\cdot}$, $i \in \mathcal{P}$; $\frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial V_{j\cdot}} = \alpha_v V_{j\cdot}$, $j \in \mathcal{A}$; $\frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial b_i} = \beta_v b_i$, $i \in \mathcal{P}$; and $\frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial b_j} = \beta_v b_j$, $j \in \mathcal{A}$.

Combining the gradients w.r.t. the loss term and the regularization term, we get the final gradients of each variable, $U_{u\cdot}$, $V_{i\cdot}$, $b_i$, $i \in \mathcal{P}$ and $V_{j\cdot}$, $b_j$, $j \in \mathcal{A}$,

$$\nabla U_{u\cdot} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial U_{u\cdot}} + \frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial U_{u\cdot}} \tag{2.5}$$

$$\nabla V_{i\cdot} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial V_{i\cdot}} + \frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial V_{i\cdot}},\ i \in \mathcal{P} \tag{2.6}$$

$$\nabla V_{j\cdot} = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial V_{j\cdot}} + \frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial V_{j\cdot}},\ j \in \mathcal{A} \tag{2.7}$$

$$\nabla b_i = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial b_i} + \frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial b_i},\ i \in \mathcal{P} \tag{2.8}$$

$$\nabla b_j = \frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial b_j} + \frac{\partial \mathcal{R}(u,\mathcal{P},\mathcal{A})}{\partial b_j},\ j \in \mathcal{A} \tag{2.9}$$

We thus have the update rules for each variable,

$$U_{u\cdot} = U_{u\cdot} - \gamma \nabla U_{u\cdot} \tag{2.10}$$

$$V_{i\cdot} = V_{i\cdot} - \gamma \nabla V_{i\cdot},\ i \in \mathcal{P} \tag{2.11}$$

$$V_{j\cdot} = V_{j\cdot} - \gamma \nabla V_{j\cdot},\ j \in \mathcal{A} \tag{2.12}$$

$$b_i = b_i - \gamma \nabla b_i,\ i \in \mathcal{P} \tag{2.13}$$

$$b_j = b_j - \gamma \nabla b_j,\ j \in \mathcal{A} \tag{2.14}$$

where $\gamma > 0$ is the learning rate.

In the SGD algorithmic framework, we approximate the objective function in Eq.(2.4) via randomly sampling one subset $\mathcal{P} \subseteq \mathcal{I}_u^{tr}$ and one subset $\mathcal{A} \subseteq \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr}$ in each iteration, instead of enumerating all possible subsets of $\mathcal{P}$ and $\mathcal{A}$. The algorithm steps of CoFiSet are depicted in Figure 2, which go through the whole data with $T$ outer loops and $n$ inner loops (one for each user on average) with $t_1$ and $t_2$ as their iteration variables, respectively. For each iteration, we first randomly sample a user $u$, and then randomly sample an item-set $\mathcal{P} \subseteq \mathcal{I}_u^{tr}$ and an item-set $\mathcal{A} \subseteq \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr}$. Once we have updated $U_{u\cdot}$, the latest $U_{u\cdot}$ is used to update $V_{i\cdot}, i \in \mathcal{P}$ and $V_{j\cdot}, j \in \mathcal{A}$. The time complexity of CoFiSet is $O(Tnd \max(|\mathcal{P}|, |\mathcal{A}|))$, where $T$ is the number of iterations, $n$ is the number of users, $d$ is the number of latent features, $|\mathcal{P}|$ is the size of item-set $\mathcal{P}$, and $|\mathcal{A}|$ is the size of item-set $\mathcal{A}$. Note that the time complexity of BPRMF [17] is $O(Tnd)$. Since $|\mathcal{P}|$ and $|\mathcal{A}|$ are usually smaller than $d$, the time complexity of CoFiSet is comparable to that of BPRMF [17], which is also supported by our empirical results of CPU time in Section 3.4.

**Input**: Training data $\mathcal{R}^{tr} = \{(u, i)\}$ of observed feedbacks, the size of item-set $\mathcal{P}$ (*presence* of observation), and the size of item-set $\mathcal{A}$ (*absence* of observation).

**Output**: The learned model parameters $\Theta = \{U_{u\cdot}, V_{i\cdot}, b_{i\cdot}, u \in \mathcal{U}^{tr}, i \in \mathcal{I}^{tr}\}$, where $U_{u\cdot}$ is the user-specific latent feature vector of user $u$, $V_{i\cdot}$ is the item-specific latent feature vector of item $i$, and $b_i$ is the bias of item $i$.

**For** $t_1 = 1, \ldots, T$**.**
  **For** $t_2 = 1, \ldots, n$**.**
  Step 1. Randomly pick a user $u \in \mathcal{U}^{tr}$.
  Step 2. Randomly pick an item-set $\mathcal{P} \subseteq \mathcal{I}_u^{tr}$.
  Step 3. Randomly pick an item-set $\mathcal{A} \subseteq \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr}$.
  Step 4. Calculate $\frac{\partial \mathcal{L}(u,\mathcal{P},\mathcal{A})}{\partial \hat{r}_{u\mathcal{P}\mathcal{A}}}$, $\bar{V}_{\mathcal{P}}$, and $\bar{V}_{\mathcal{A}}$.
  Step 5. Update $U_{u\cdot}$ via Eq.(2.5, 2.10).
  Step 6. Update $V_{i\cdot}, i \in \mathcal{P}$ via Eq.(2.6, 2.11) and the latest $U_{u\cdot}$.
  Step 7. Update $V_{j\cdot}, j \in \mathcal{A}$ via Eq.(2.7, 2.12) and the latest $U_{u\cdot}$.
  Step 8. Update $b_i, i \in \mathcal{P}$ via Eq.(2.8, 2.13).
  Step 9. Update $b_j, j \in \mathcal{A}$ via Eq.(2.9, 2.14).
  **End**
**End**

Figure 2: The algorithm of **co**llaborative **fi**ltering via learning pairwise preferences over item-**set**s (CoFiSet).

**2.5 Discussion** For the loss term $\mathcal{L}(u, \mathcal{P}, \mathcal{A})$ in Eq.(2.4), we can have various specific forms, e.g.

$-\ln\sigma(\hat{r}_{u\mathcal{P}\mathcal{A}})$, $\frac{1}{2}(\hat{r}_{u\mathcal{P}\mathcal{A}}-1)^2$, and $\max(0,1-\hat{r}_{u\mathcal{P}\mathcal{A}})$, where $\hat{r}_{u\mathcal{P}\mathcal{A}}=\hat{r}_{u\mathcal{P}}-\hat{r}_{u\mathcal{A}}$ is the difference of user $u$'s preferences on two item-sets $\mathcal{P}$ and $\mathcal{A}$, and $\sigma(z)=1/(1+\exp(-z))$ is the sigmoid function. The loss $-\ln\sigma(\hat{r}_{u\mathcal{P}\mathcal{A}})$ absorbs that of BPRMF [17] as a special case when $\mathcal{P}=\{i\}$ and $\mathcal{A}=\{j\}$, $\mathcal{L}(u,\mathcal{P},\mathcal{A})=\mathcal{L}(u,\{i\},\{j\})=-\ln\sigma(\hat{r}_{uij})$; the loss $\frac{1}{2}(\hat{r}_{u\mathcal{P}\mathcal{A}}-1)^2$ absorbs that of RankALS [19] as a special case when $\mathcal{P}=\{i\}$ and $\mathcal{A}=\{j\}$, $\mathcal{L}(u,\mathcal{P},\mathcal{A})=\mathcal{L}(u,\{i\},\{j\})=\frac{1}{2}(\hat{r}_{uij}-1)^2$; and the loss $\max(0,1-\hat{r}_{u\mathcal{P}\mathcal{A}})$ absorbs that of CCF(Hinge) [20] as a special case when $\mathcal{P}=\{i\}$, $\mathcal{L}(u,\mathcal{P},\mathcal{A})=\mathcal{L}(u,\{i\},\mathcal{A})=\max(0,1-\hat{r}_{ui\mathcal{A}})$.

We can thus see that our proposed optimization framework in Eq.(2.4) is quite general and able to absorb BPRMF [17], RankALS [19] and CCF(Hinge) [20] as special cases. For CoFiSet, we use the loss term $-\ln\sigma(\hat{r}_{u\mathcal{P}\mathcal{A}})$, since then we can more directly compare our *pairwise preferences over item-sets* with *pairwise preferences over items* made in BPRMF [17].

## 3 Experimental Results

**3.1 Data Sets** We use two real-world data sets, MovieLens100K[1] and Epinions-Trustlet[2] [12], to empirically study our assumption of *pairwise preferences over item-sets*. For MovieLens100K, we keep ratings larger than 3 as observed feedbacks [3]. For Epinions-Trustlet, we keep users with at least 25 social connections [18]. Finally, we have 55375 observations from 942 users and 1447 items in MovieLens100K, and 346035 observations from 4718 users and 36165 items in Epinions-Trustlet. In our experiments, for each user, we randomly take 50% of the corresponding observed feedbacks as training data and the rest 50% as test data. We repeat this for 5 times to generate 5 copies of training data and test data, and report the average performance on those 5 copies of data.

**3.2 Evaluation Metrics** Once we have learned the model parameters, we can calculate the prediction score for user $u$ on item $i$, $\hat{r}_{ui}=U_{u\cdot}V_{i\cdot}^T+b_i$, and then get a ranking list, $i(1),\ldots,i(\ell),\ldots,i(k),\ldots$, where $i(\ell)$ represents the item located at position $\ell$. For each item $i$, we can also have its position $1\le p_{ui}\le m$.

We study the recommendation performance on various commonly used ranking-oriented evaluation metrics, $Pre@k$ [18, 20], $NDCG@k$ [20], MRR (mean reciprocal rank) [18], ARP (average relative position) [19], and AUC (area under the curve) [17].

1. **Pre@$k$** The precision of user $u$ is defined as, $Pre_u@k=\frac{1}{k}\sum_{\ell=1}^{k}\delta(i(\ell)\in\mathcal{I}_u^{te})$, where $\delta(x)=1$ if $x$ is true and $\delta(x)=0$ otherwise. $\sum_{\ell=1}^{k}\delta(i(\ell)\in\mathcal{I}_u^{te})$ thus denotes the number of items among the top-$k$ recommended items that have observed feedbacks from user $u$. Then, we have $Pre@k=\sum_{u\in\mathcal{U}^{te}}Pre_u@k/|\mathcal{U}^{te}|$.
2. **NDCG@$k$** The NDCG of user $u$ is defined as, $NDCG_u@k=\frac{1}{Z_u}DCG_u@k$, with $DCG_u@k=\sum_{\ell=1}^{k}\frac{2^{\delta(i(\ell)\in\mathcal{I}_u^{te})}-1}{log(\ell+1)}$, where $Z_u$ is the best $DCG_u@k$ score. Then, we have $NDCG@k=\sum_{u\in\mathcal{U}^{te}}NDCG_u@k/|\mathcal{U}^{te}|$.
3. **MRR** The reciprocal rank of user $u$ is defined as, $RR_u=\frac{1}{\min_{i\in\mathcal{I}_u^{te}}(p_{ui})}$, where $\min_{i\in\mathcal{I}_u^{te}}(p_{ui})$ is the position of the first relevant item in the estimated ranking list for user $u$. Then, we have $MRR=\sum_{u\in\mathcal{U}^{te}}RR_u/|\mathcal{U}^{te}|$.
4. **ARP** The relative position of user $u$ is defined as, $RP_u=\frac{1}{|\mathcal{I}_u^{te}|}\sum_{i\in\mathcal{I}_u^{te}}\frac{p_{ui}}{|\mathcal{I}^{tr}|-|\mathcal{I}_u^{tr}|}$, where $\frac{p_{ui}}{|\mathcal{I}^{tr}|-|\mathcal{I}_u^{tr}|}$ is the relative position of item $i$. Then, we have $ARP=\sum_{u\in\mathcal{U}^{te}}RP_u/|\mathcal{U}^{te}|$.
5. **AUC** The AUC of user $u$ is defined as, $AUC_u=\frac{1}{|\mathcal{R}^{te}(u)|}\sum_{(i,j)\in\mathcal{R}^{te}(u)}\delta(\hat{r}_{ui}>\hat{r}_{uj})$, where $\mathcal{R}^{te}(u)=\{(i,j)|(u,i)\in\mathcal{R}^{te},(u,j)\notin\mathcal{R}^{tr}\cup\mathcal{R}^{te}\}$. Then, we have $AUC=\sum_{u\in\mathcal{U}^{te}}AUC_u/|\mathcal{U}^{te}|$.

In the evaluation of Epinions-Trustlet, we ignore the most popular three items (or trustees in the social network) in the recommended list [18], in order to alleviate the domination effect from those items.

**3.3 Baselines and Parameter Settings** We compare CoFiSet with several state-of-the-art methods, including RankALS [19], CCF(Hinge) [20], CCF(SoftMax) [20], and BPRMF [17]. We also compare to a commonly used baseline called PopRank (ranking via popularity of the items) [18].

For fair comparison, we implement RankALS, CCF(Hinge), CCF(SoftMax), BPRMF and CoFiSet in the same code framework written in Java, and use the same initializations for the model variables, $U_{uk}=(r-0.5)\times 0.01, k=1,\ldots,d$, $V_{ik}=(r-0.5)\times 0.01, k=1,\ldots,d$, $b_i=\sum_{u\in\mathcal{U}_i^{tr}}1/n-\sum_{(u,i)\in\mathcal{R}^{tr}}1/n/m$, where $r$ $(0\le r<1)$ is a random value. The order of updating the variables in each iteration is also the same as that shown in Figure 2. Note that we can use the initialization of item bias, $b_i$, to rank the items, which is actually PopRank.

For the iteration number $T$, we tried $T\in\{10^4,10^5,10^6\}$ for all methods on MovieLens100K (when $d=10$), and found that the results using $T\in$

[1] http://www.grouplens.org/
[2] http://www.trustlet.org/wiki/Downloaded_Epinions_dataset

$\{10^5, 10^6\}$ are similar and much better than that of using $T = 10^4$. We thus fix $T = 10^5$. For the number of latent features, we use $d \in \{10, 20\}$ [3, 18]. For the tradeoff parameters, we search the best values from $\alpha_u = \alpha_v = \beta_v \in \{0.001, 0.01, 0.1\}$ using $NDCG@5$ performance on the first copy of data, and then fix them in the rest four copies of data [18]. We find that the best values of the tradeoff parameters for different models on different data sets can be different, which are reported in Table 2 and Table 3. The learning rate is fixed as $\gamma = 0.01$.

For CCF(Hinge), we use $1/(1+\exp[-100(1-\hat{r}_{ui\mathcal{A}})])$ as suggested by [20] to approximate $\delta(1 - \hat{r}_{ui\mathcal{A}} > 0)$ for the non-smooth issue of Hinge loss. For CCF(Hinge) and CCF(SoftMax), we use $|\mathcal{A}| = 2$. Note that for CCF(Hinge) and CCF(SoftMax), there is no item-set $\mathcal{P}$. For CoFiSet, we first fix $|\mathcal{P}| = 2$ and $|\mathcal{A}| = 1$ (to be fair with CCF), and then try different values of $|\mathcal{P}| \in \{1, 2, 3, 4, 5\}$ and $|\mathcal{A}| \in \{1, 2, 3, 4, 5\}$. In the case that there are not enough observed feedbacks in $\mathcal{I}_u^{tr}$, we use $\mathcal{P} = \mathcal{I}_u^{tr}$.

### 3.4 Summary of Experimental Results

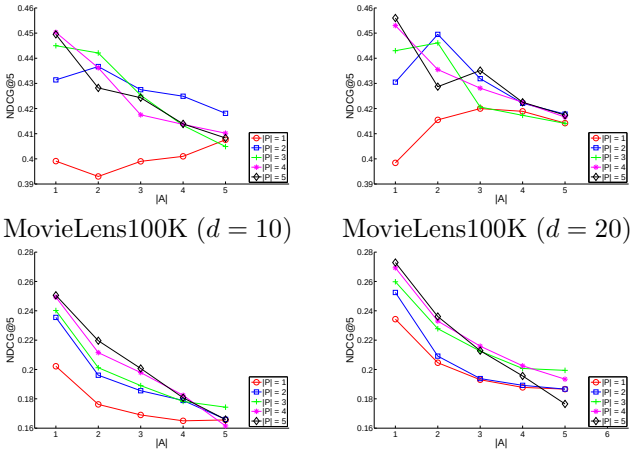
The prediction performance of CoFiSet and baselines are shown in Table 2 and Table 3, from which we can have the following observations,

1. For both data sets, CoFiSet achieves better performance than all baselines in most cases. The result clearly demonstrates the superior prediction ability of CoFiSet. More importantly, the improvements on top-$k$ related metrics are even more significant, which have been known to be critical for a real recommender system, since users usually only check a few items which are ranked in top positions [1].

2. For baselines, we can see that all algorithms beat PopRank, which shows the effectiveness of the preference assumptions made in RankALS, CCF(Hinge), CCF(SoftMax) and BPRMF, though their results are still worse than that of CoFiSet.

3. For the two closely related competitive baselines, BPRMF performs better than CCF(SoftMax) when $d = 10$ in MovieLens100K regarding $ARP$ and $AUC$, but worse in other cases. The performance is similar in Epinions-Trustlet. On the contrary, CoFiSet performs stable on both data sets, which again shows the effectiveness of our relaxed assumption of *pairwise preferences over item-sets* in CoFiSet, relative to *pairwise preferences over items* in BPRMF [17].

We then study CoFiSet with different sizes of item-sets $\mathcal{P}$ and $\mathcal{A}$. For each combination of $|\mathcal{P}|$ and $|\mathcal{A}|$, the best values of tradeoff parameters, $\alpha_u = \alpha_v = \beta_v \in \{0.001, 0.01, 0.1\}$, are searched in the same way. The results of $NDCG@5$ on MovieLens100K and Epinions-Trustlet are shown in Figure 3. The main findings are,




MovieLens100K ($d = 10$) MovieLens100K ($d = 20$)

Epinions-Trustlet ($d = 10$) Epinions-Trustlet ($d = 20$)

Figure 3: Prediction performance of CoFiSet with different sizes of item-set $\mathcal{P}$ and item-set $\mathcal{A}$. We fix $T = 10^5$. Note that when $|\mathcal{P}| = |\mathcal{A}| = 1$, CoFiSet reduces to BPRMF [17].

1. The best performance locates in the left-top corner in each sub-figure, which shows that CoFiSet prefers a relatively large value of $|\mathcal{P}|$ and small value of $|\mathcal{A}|$. This result is quite interesting and is different from that of CCF [20], which uses a relatively large item-set $\mathcal{A}$ instead. This phenomenon can be explained by the fact that there is a higher chance to have inconsistent preferences on items from item-set $\mathcal{A}$ than from item-set $\mathcal{P}$. Hence, in practice, we may use a relatively large item-set $\mathcal{P}$ and a small item-set $\mathcal{A}$ in CoFiSet as a guideline.

2. When $|\mathcal{P}| = |\mathcal{A}| = 1$, CoFiSet reduces to BPRMF. We can thus again see the advantages of our assumption comparing to that in BPRMF [17].

We also study the efficiency of CoFiSet with different values of $|\mathcal{P}|$ and $|\mathcal{A}|$, which is shown in Figure 4. We can see that (1) the time cost is almost linear w.r.t. the value of $|\mathcal{A}|$ given $|\mathcal{P}|$, and vise versa, and (2) CoFiSet is very efficient since both CoFiSet and BPRMF are of the same order of CPU time. This result is consistent to the analysis of time complexity of CoFiSet and BPRMF in Section 2.4.

Table 2: Prediction performance on MovieLens100K of PopRank, RankALS [19] implemented in the SGD framework, CCF(Hinge) [20], CCF(SoftMax) [20], BPRMF [17], and CoFiSet. We fix $|\mathcal{P}| = 2$ and $|\mathcal{A}| = 1$ for CoFiSet, and fix $|\mathcal{A}| = 2$ for CCF(Hinge) and CCF(SoftMax). The up arrow $\uparrow$ means the larger the better of the results on the corresponding metric, and the down arrow $\downarrow$ the smaller the better. Numbers in boldface (e.g. **0.4112**) are the best results, and numbers in Italic (e.g. *0.3983*) are the second best results. The best values of tradeoff parameters ($\alpha_u = \alpha_v = \beta_v$) for different models are also included for reference.

| | $\alpha_u, \alpha_v, \beta_v$ | $Pre@5 \uparrow$ | $NDCG@5 \uparrow$ | $MRR \uparrow$ | $ARP \downarrow$ | $AUC \uparrow$ |
|---|---|---|---|---|---|---|
| PopRank | N/A | $0.2687_{\pm 0.0040}$ | $0.2900_{\pm 0.0033}$ | $0.5079_{\pm 0.0074}$ | $0.1532_{\pm 0.0011}$ | $0.8544_{\pm 0.0012}$ |

| $d = 10$ | $\alpha_u, \alpha_v, \beta_v$ | $Pre@5 \uparrow$ | $NDCG@5 \uparrow$ | $MRR \uparrow$ | $ARP \downarrow$ | $AUC \uparrow$ |
|---|---|---|---|---|---|---|
| RankALS(SGD) | 0.1 | $0.3836_{\pm 0.0086}$ | $0.3975_{\pm 0.0123}$ | $0.6019_{\pm 0.0215}$ | $0.0925_{\pm 0.0014}$ | $0.9161_{\pm 0.0015}$ |
| CCF(Hinge) | 0.1 | $0.3806_{\pm 0.0053}$ | $0.3947_{\pm 0.0116}$ | $0.5984_{\pm 0.0232}$ | $\mathit{0.0903}_{\pm 0.0015}$ | $\mathit{0.9183}_{\pm 0.0015}$ |
| CCF(SoftMax) | 0.1 | $\mathit{0.3983}_{\pm 0.0028}$ | $\mathit{0.4194}_{\pm 0.0017}$ | $\mathit{0.6357}_{\pm 0.0056}$ | $0.0934_{\pm 0.0014}$ | $0.9151_{\pm 0.0014}$ |
| BPRMF | 0.01 | $0.3823_{\pm 0.0052}$ | $0.3991_{\pm 0.0060}$ | $0.6065_{\pm 0.0068}$ | $0.0917_{\pm 0.0013}$ | $0.9169_{\pm 0.0013}$ |
| CoFiSet | 0.01 | $\mathbf{0.4112}_{\pm 0.0066}$ | $\mathbf{0.4314}_{\pm 0.0085}$ | $\mathbf{0.6399}_{\pm 0.0140}$ | $\mathbf{0.0884}_{\pm 0.0010}$ | $\mathbf{0.9203}_{\pm 0.0010}$ |

| $d = 20$ | $\alpha_u, \alpha_v, \beta_v$ | $Pre@5 \uparrow$ | $NDCG@5 \uparrow$ | $MRR \uparrow$ | $ARP \downarrow$ | $AUC \uparrow$ |
|---|---|---|---|---|---|---|
| RankALS(SGD) | 0.1 | $0.3906_{\pm 0.0035}$ | $0.4043_{\pm 0.0090}$ | $0.6071_{\pm 0.0203}$ | $0.0931_{\pm 0.0017}$ | $0.9154_{\pm 0.0017}$ |
| CCF(Hinge) | 0.1 | $\mathit{0.3993}_{\pm 0.0066}$ | $\mathit{0.4186}_{\pm 0.0074}$ | $0.6296_{\pm 0.0094}$ | $\mathbf{0.0901}_{\pm 0.0014}$ | $\mathbf{0.9185}_{\pm 0.0015}$ |
| CCF(SoftMax) | 0.1 | $0.3955_{\pm 0.0063}$ | $0.4185_{\pm 0.0062}$ | $\mathit{0.6389}_{\pm 0.0117}$ | $0.0934_{\pm 0.0015}$ | $0.9151_{\pm 0.0015}$ |
| BPRMF | 0.1 | $0.3772_{\pm 0.0101}$ | $0.3984_{\pm 0.0102}$ | $0.6165_{\pm 0.0122}$ | $0.1032_{\pm 0.0017}$ | $0.9050_{\pm 0.0017}$ |
| CoFiSet | 0.01 | $\mathbf{0.4104}_{\pm 0.0083}$ | $\mathbf{0.4305}_{\pm 0.0098}$ | $\mathbf{0.6421}_{\pm 0.0137}$ | $\mathit{0.0915}_{\pm 0.0019}$ | $\mathit{0.9170}_{\pm 0.0019}$ |

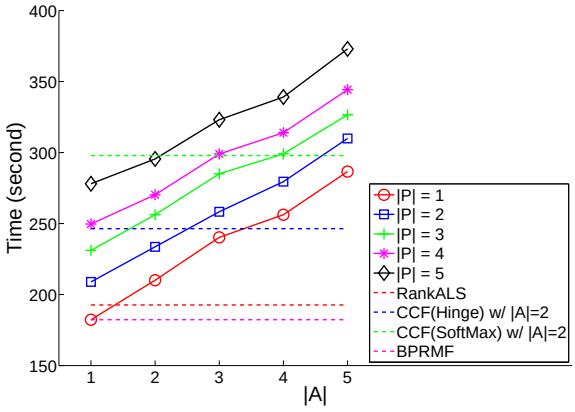



Figure 4: The CPU time on training CoFiSet with different values of $|\mathcal{P}|$ and $|\mathcal{A}|$ and baselines on MovieLens100K. We fix $T = 10^5$ and $d = 10$. The experiments are conducted on Linux research machines with Xeon X5570 @ 2.93GHz(2-CPU/4-core) / 32GB RAM / 32GB SWAP, and Xeon X5650 @ 2.67GHz (2-CPU/6-core) / 32GB RAM / 32GB SWAP.

## 4 Related Work

In this section, we discuss some closely related algorithms in collaborative filtering with implicit feedbacks.

CLiMF (collaborative less-is-more filtering) [18] proposes to encourage self-competitions among observed items only via maximizing $\sum_{i \in \mathcal{I}_u^{tr}} \left[ \ln \sigma(\hat{r}_{ui}) + \sum_{i' \in \mathcal{I}_u^{tr} \setminus \{i\}} \ln \sigma(\hat{r}_{ui} - \hat{r}_{ui'}) \right]$ for each user $u$. The unobserved items from $\mathcal{I}^{tr} \setminus \mathcal{I}_u^{tr}$ are ignored, which may miss some information during model training.

iMF (implicit matrix factorization) [6] and OCCF (one-class collaborative filtering) [14] propose to minimize $\sum_{i \in \mathcal{I}_u^{tr}} c_{ui}(1 - \hat{r}_{ui})^2 + \sum_{j \in \mathcal{I}^{tr} \setminus \mathcal{I}_u^{tr}} c_{uj}(0 - \hat{r}_{uj})^2$ for each user $u$, where $c_{ui}$ and $c_{uj}$ are estimated confidence values [6, 14]. We can see that this objective function is based on *pointwise preferences on items*, which is empirically to be less competitive than pairwise preferences [17].

BPRMF (Bayesian personalized ranking based matrix factorization) [17] proposes a relaxed assumption of *pairwise preferences over items*, and minimizes $\sum_{i \in \mathcal{I}_u^{tr}} \sum_{j \in \mathcal{I}^{tr} \setminus \mathcal{I}_u^{tr}} -\ln \sigma(\hat{r}_{uij})$ for each user $u$. The difference of user $u$'s preferences on items $i$ and $j$, $\hat{r}_{uij} = \hat{r}_{ui} - \hat{r}_{uj}$, is a special case of that in CoFiSet. In some recommender system like LinkedIn[3], a user $u$ may click more than one social updates

[3] http://www.linkedin.com/

Table 3: Prediction performance on Epinions-Trustlet of PopRank, RankALS [19] implemented in the SGD framework, CCF(Hinge) [20], CCF(SoftMax) [20], BPRMF [17], and CoFiSet. We fix $|\mathcal{P}| = 2$ and $|\mathcal{A}| = 1$ for CoFiSet, and fix $|\mathcal{A}| = 2$ for CCF(Hinge) and CCF(SoftMax). The up arrow $\uparrow$ means the larger the better of the results on the corresponding metric, and the down arrow $\downarrow$ the smaller the better. Numbers in boldface (e.g. **0.2254**) are the best results, and numbers in Italic (e.g. *0.2014*) are the second best results. The best values of tradeoff parameters ($\alpha_u = \alpha_v = \beta_v$) for different models are also included for reference.

| | $\alpha_u, \alpha_v, \beta_v$ | $Pre@5 \uparrow$ | $NDCG@5 \uparrow$ | $MRR \uparrow$ | $ARP \downarrow$ | $AUC \uparrow$ |
|---|---|---|---|---|---|---|
| PopRank | N/A | $0.0837_{\pm 0.0018}$ | $0.0848_{\pm 0.0019}$ | $0.2022_{\pm 0.0027}$ | $0.1270_{\pm .0004}$ | $0.8734_{\pm 0.0004}$ |

| $d = 10$ | $\alpha_u, \alpha_v, \beta_v$ | $Pre@5 \uparrow$ | $NDCG@5 \uparrow$ | $MRR \uparrow$ | $ARP \downarrow$ | $AUC \uparrow$ |
|---|---|---|---|---|---|---|
| RankALS(SGD) | 0.1 | $0.1283_{\pm 0.0104}$ | $0.1305_{\pm 0.0116}$ | $0.2754_{\pm 0.0182}$ | $0.0962_{\pm 0.0002}$ | $0.9042_{\pm 0.0002}$ |
| CCF(Hinge) | 0.1 | $0.1499_{\pm 0.0033}$ | $0.1532_{\pm 0.0029}$ | $0.3073_{\pm 0.0065}$ | $0.0956_{\pm 0.0003}$ | $0.9048_{\pm 0.0003}$ |
| CCF(SoftMax) | 0.1 | $\mathit{0.2014}_{\pm 0.0059}$ | $\mathit{0.2089}_{\pm 0.0065}$ | $\mathit{0.3829}_{\pm 0.0087}$ | $0.0968_{\pm 0.0002}$ | $0.9036_{\pm 0.0002}$ |
| BPRMF | 0.01 | $0.1964_{\pm 0.0033}$ | $0.2022_{\pm 0.0042}$ | $0.3639_{\pm 0.0061}$ | $\mathit{0.0920}_{\pm 0.0003}$ | $\mathit{0.9084}_{\pm 0.0003}$ |
| CoFiSet | 0.01 | $\mathbf{0.2254}_{\pm 0.0025}$ | $\mathbf{0.2355}_{\pm 0.0024}$ | $\mathbf{0.4166}_{\pm 0.0020}$ | $\mathbf{0.0913}_{\pm 0.0004}$ | $\mathbf{0.9091}_{\pm 0.0004}$ |

| $d = 20$ | $\alpha_u, \alpha_v, \beta_v$ | $Pre@5 \uparrow$ | $NDCG@5 \uparrow$ | $MRR \uparrow$ | $ARP \downarrow$ | $AUC \uparrow$ |
|---|---|---|---|---|---|---|
| RankALS(SGD) | 0.1 | $0.1437_{\pm 0.0040}$ | $0.1473_{\pm 0.0052}$ | $0.3028_{\pm 0.0096}$ | $0.0967_{\pm 0.0002}$ | $0.9037_{\pm 0.0002}$ |
| CCF(Hinge) | 0.1 | $0.1735_{\pm 0.0018}$ | $0.1765_{\pm 0.0013}$ | $0.3361_{\pm 0.0027}$ | $0.0962_{\pm 0.0002}$ | $0.9042_{\pm 0.0002}$ |
| CCF(SoftMax) | 0.01 | $\mathit{0.2299}_{\pm 0.0027}$ | $\mathit{0.2353}_{\pm 0.0031}$ | $0.4028_{\pm 0.0051}$ | $0.0922_{\pm 0.0004}$ | $0.9082_{\pm 0.0004}$ |
| BPRMF | 0.01 | $0.2279_{\pm 0.0033}$ | $0.2343_{\pm 0.0031}$ | $\mathit{0.4044}_{\pm 0.0043}$ | $\mathit{0.0916}_{\pm 0.0004}$ | $\mathit{0.9088}_{\pm 0.0004}$ |
| CoFiSet | 0.01 | $\mathbf{0.2438}_{\pm 0.0034}$ | $\mathbf{0.2525}_{\pm 0.0042}$ | $\mathbf{0.4332}_{\pm 0.0067}$ | $\mathbf{0.0912}_{\pm 0.0003}$ | $\mathbf{0.9092}_{\pm 0.0003}$ |

(or items) in one single impression (or session), and PLMF (pairwise learning via matrix factorization) [5] adopts a similar idea of BPRMF [17] and minimizes $\frac{1}{|\mathcal{O}_{us}^+||\mathcal{O}_{us}^-|}\sum_{i\in\mathcal{O}_{us}^+}\sum_{j\in\mathcal{O}_{us}^-} -\sigma(\hat{r}_{uij})$, where $\mathcal{O}_{us}^+$ and $\mathcal{O}_{us}^-$ are sets of clicked and un-clicked items, respectively, by user $u$ in session $s$ . We can see that the pairwise preference is also defined on clicked and un-clicked items instead of item-sets as used in CoFiSet.

RankALS (ranking-based alternative least square) [19] adopts a square loss and minimizes $\sum_{i\in\mathcal{I}_u^{tr}}\sum_{j\in\mathcal{I}^{tr}\setminus\mathcal{I}_u^{tr}}\frac{1}{2}(\hat{r}_{uij}-1)^2$ for each user, where $\hat{r}_{uij}$ is again the user $u$'s pairwise preferences over items $i$ and $j$. Note that RankALS is motivated by incorporating the preference difference on two items [7], $\hat{r}_{ui}-\hat{r}_{uj}=1$, into the ALS (alternative least square) [6] algorithmic framework, and optimizes a slightly different objective function. In our experiments, for fair comparison, we implement it in the SGD framework, which is the same as for other baselines.

CCF(SoftMax) [20] assumes that there is a candidate set $\mathcal{O}_{ui}$ for each observed pair $(u,i)$, which can be written as $\mathcal{O}_{ui} = \{i\} \cup \mathcal{A}$. CCF(SoftMax) models the data as a competitive game and proposes to minmize $-\ln\frac{\exp(\hat{r}_{ui})}{\exp(\hat{r}_{ui})+\sum_{j\in\mathcal{A}}\exp(\hat{r}_{uj})} = \ln[1+\sum_{j\in\mathcal{A}}\exp(-\hat{r}_{uij})]$ for each observed pair $(u,i)$, where $\hat{r}_{uij} = \hat{r}_{ui} - \hat{r}_{uj}$. We can see that CCF(SoftMax) defines the loss on pairwise preferences over items instead of item-sets, which is thus different from our CoFiSet. Note that when $\mathcal{A} = \{j\}$, the loss term of CCF(SoftMax) reduces to that of BPRMF [17], which is a special case of CoFiSet.

CCF(Hinge) [20] adopts a Hinge loss over an item $i$ and an item-set $\mathcal{O}_{ui}\setminus\{i\} = \mathcal{A}$ for each observed pair $(u,i)$, and minimizes $\max(0, 1-\hat{r}_{ui\mathcal{A}})$, where $\hat{r}_{ui\mathcal{A}} = \hat{r}_{ui} - \hat{r}_{u\mathcal{A}}$ with $\hat{r}_{u\mathcal{A}} = \sum_{j\in\mathcal{A}}\hat{r}_{uj}/|\mathcal{A}|$. We can see that the loss term of CCF(Hinge) [20] can be considered as a special case in our CoFiSet when $\mathcal{P} = \{i\}$.

The above related works are summarized in Table 4. From Table 4 and discussions above, we can see that (1) CoFiSet is different from other algorithms, since it is based on a new assumption of *pairwise preferences over item-sets*, and (2) the most closely related works are BPRMF [17], CCF(SoftMax) [20] and PLMF [5], because they also adopt pairwise preference assumptions, exponential family functions in loss terms, and SGD (stochastic gradient descent) style algorithms.

## 5 Conclusions and Future Work

In this paper, we propose a novel algorithm, CoFiSet, in collaborative filtering with implicit feedbacks. Specifi-

Table 4: Summary of CoFiSet and some related works in collaborative filtering with implicit feedbacks. Note that $i, i' \in \mathcal{I}_u^{tr}$, $j \in \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr}$, $\mathcal{P} \subseteq \mathcal{I}_u^{tr}$, and $\mathcal{A} \subseteq \mathcal{I}^{tr} \backslash \mathcal{I}_u^{tr}$. The relationship "$x$ v.s. $y$" denotes encouragement of competitions between $x$ and $y$, "$x - y = c$" means that the difference between a user's preferences on $x$ and $y$ is a constant, and "$x \succ y$" means that a user prefers $x$ to $y$.

| Preference type/assumption | | Algorithm | |
|---|---|---|---|
| *Self-competition* | $i$ v.s. $i'$ | *CLiMF* [18] | Batch |
| *Pointwise* | $i$: like | *iMF* [6] | Batch |
| | $j$: dislike | *OCCF* [14] | Batch |
| *Pairwise* | $i - j = c$ | *RankALS* [19] | Batch |
| | | *SVD(ranking)* [7] | SGD |
| | $i \succ j$ | *BPRMF* [17] | SGD |
| | | *PLMF* [5] | SGD |
| | $i \succ j, j \in \mathcal{A}$ | *CCF(SoftMax)* [20] | SGD |
| | $i \succ \mathcal{A}$ | *CCF(Hinge)* [20] | SGD |
| | $\mathcal{P} \succ \mathcal{A}$ | ***CoFiSet*** | SGD |

cally, we propose a new assumption, *pairwise preferences over item-sets*, which is more relaxed than *pairwise preferences over items* in previous works. With this assumption, we develop a general algorithm, which absorbs some recent algorithms as special cases. We study CoFiSet on two real-world data sets using various ranking-oriented evaluation metrics, and find that CoFiSet generates better recommendations than several state-of-the-art methods. CoFiSet works best especially when it is associated with a small item-set $\mathcal{A}$, because there is a higher chance to have inconsistent preferences on items from item-set $\mathcal{A}$.

For future works, we are mainly interested in extending CoFiSet in three aspects, (1) studying item-set selection strategies via incorporating the item's taxonomy information, (2) modeling different preference assumptions in a unified ranking-oriented framework, and (3) applying the concept of *item-set* to other matrix or tensor factorization algorithms.

## 6 Acknowledgments

This research work was partially supported by Hong Kong Research Grants Council under project ECS/HKBU211912.

## References

[1] Li Chen and Pearl Pu. Users' eye gaze pattern in organization-based recommender interfaces. In *IUI*, 2011.

[2] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. The youtube video recommendation system. In *RecSys*, 2010.

[3] Liang Du, Xuan Li, and Yi-Dong Shen. User graph regularized pairwise matrix factorization for item recommendation. In *ADMA*, 2011.

[4] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *CACM*, 35(12), 1992.

[5] Liangjie Hong, Ron Bekkerman, Joseph Adler, and Brian D. Davison. Learning to rank social update streams. In *SIGIR*, 2012.

[6] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, 2008.

[7] Michael Jahrer and Andreas Toscher. Collaborative filtering ensemble for ranking. In *KDDCUP*, 2011.

[8] Micheline Kamber Jiawei Han and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2011.

[9] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*, 2008.

[10] Bin Li, Qiang Yang, and Xiangyang Xue. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, 2009.

[11] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE IC*, 7(1), 2003.

[12] Paolo Massa and Paolo Avesani. Trust-aware bootstrapping of recommender systems. In *ECAI Workshop on Recommender Systems*, 2006.

[13] Xia Ning and George Karypis. Slim: Sparse linear methods for top-n recommender systems. In *ICDM*, 2011.

[14] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *ICDM*, 2008.

[15] Weike Pan, Evan Wei Xiang, and Qiang Yang. Transfer learning in collaborative filtering with uncertain ratings. In *AAAI*, 2012.

[16] Steffen Rendle. Factorization machines with libfm. *ACM TIST*, 3(3), 2012.

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.

[18] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *RecSys*, 2012.

[19] Gábor Takács and Domonkos Tikk. Alternating least squares for personalized ranking. In *RecSys*, 2012.

[20] Shuang-Hong Yang, Bo Long, Alexander J. Smola, Hongyuan Zha, and Zhaohui Zheng. Collaborative competitive filtering: learning recommender using context of user choice. In *SIGIR*, 2011.