# Adaptive Pairwise Learning for Personalized Ranking with Content and Implicit Feedback

***Abstract*—Pairwise learning algorithms are a vital technique for personalized ranking with implicit feedback. They usually assume that each user is more interested in items which have been selected by the user than remaining ones. This pairwise assumption usually derives massive training pairs. To deal with such large-scale training data, the learning algorithms are usually based on stochastic gradient descent with uniformly drawn pairs. However, the uniformly sampling strategy often results in slow convergence. In this paper, we first uncover the reasons of slow convergence. Then, we associate contents of entities with characteristics of data sets to develop an adaptive item sampler for drawing informative training data. In this end, to devise a robust personalized ranking method, we accordingly embed our sampler into Bayesian Personalized Ranking (BPR) framework, and further propose a Content-aware and Adaptive Bayesian Personalized Ranking (CA-BPR) method, which can model both contents and implicit feedbacks in a unified learning process. The experimental results show that, our adaptive item sampler indeed can speed up BPR learning and CA-BPR definitively outperforms the state-of-the-art methods in personalized ranking.**

***Keywords*—*Personalized Ranking; Adaptive Sampling; Pairwise Learning***

## I. Introduction

Due to information overload in the Web, recommender systems with personalized ranking have become an important part in modern applications. For example, Taobao and Amazon embed personalized ranking techniques into their recommender systems in order to exactly direct potential customers to products. Most of studies [1] [2] [3] in personalized ranking are designed for explicit ratings. However, explicit ratings are generated by users actively interacting with the systems, and are hard to be obtained in practice. For instance, MovieLens needs users to rate movies on its website, and then personalized recommendation is provided based on these explicit ratings. For real-world recommender systems, we scale our research data to implicit feedbacks, such as whether a user has browsed a web page, or whether a customer has purchased a product. Such binary signals are more widespread and are easier to be collected by recommender systems than explicit ratings.

Pairwise learning algorithms, such as Bayesian Personalized Ranking (BPR) [4] and its extensions [5] [6] [3], are tailored to personalized ranking with implicit feedback. They usually assume that users are more interested in items that they have selected than the remaining items, and amount of training pairs will be derived. For dealing with large-scale training data, the learning of parameters is typically based on the stochastic gradient descent (SGD) with uniformly drawn pairs. However, the uniformly sampling strategy may result in slow convergence, especially when the number of items is large. In practice, each user has only seen a handful of items, and has provided feedbacks on some of these viewed items (these feedbacks indicate the user may like these items). Uniformly sampling strategy equally draws negative items from the total set of items. Massive meaningless training pairs are produced by this kind of sampling strategy, and these training data can not provide helpful information to tune the personalized ranking lists. For example, a smart phone and a toothbrush may be drawn to be a training pair. Intuitively, we can not say that a user dislikes a toothbrush because he likes a smart phone, this pair is meaningless in real-world scenarios. Moreover, uniformly sampling strategy draws a lot of items which are impossible to be viewed by related users. Since a user in fact is no chance to assess an unseen item, it is hard to correctly estimate the user's preference on the unseen item, and the pairs containing unseen items can not provide useful information to tune the personalized ranking lists. For instance, when a user searches smart phones on a website, the smart phones of non-mainstream brands, usually are ranked at tail places in result list by the retrieval system of the website, and the user tends to miss them. Thus, we are hard to accurately assess how much the user like a non-mainstream smart phone, and its corresponding training sample usually has low contribution to the learning of parameters. Generally speaking, different training pairs have different effect on the learning of parameters. How to utilize the properties of data sets, e.g., the distribution of user activity and item popularity, and other auxiliary information, e.g., contents of entities, to devise a more efficient sampling strategy for pairwise learning is a thorny problem.

To the best of our knowledge, there are two directions of improving uniformly sampling strategy, i.e., directly utilizing distribution of data set, e.g., long-tailed distribution, to draw training data [7], or leveraging strategies [7] [8] to automatically approximate realistic distribution of data set and adaptively draw training data. The methods of former direction usually rely on much of prior knowledge. Since different data sets may have different kinds of data distributions, these methods are hard to be scaled to general applications. Thus, we address the problem of slow convergence with developing an adaptive item sampler. Note that our sampler performs better than previous work [7] [8] in terms of balance of efficiency and performance. It holds a self-evident rule, i.e., "apples to apples", for drawing comparable items for each observed user-item pair. For example, suppose that a user has bought an iPhone. For this observed user-item pair, we tend to draw other kinds of smart phones which are comparable to iPhone in terms of product attributes, rather than other things, such as toothbrushes, which are not under the category of smart phone. Moreover, under the category of smart phone, we further select those products which probably have been viewed by the user.

With increasing of social media, apart from collaborative information, there is much content information, such as posts

of users and descriptions of items. Using content information, we also can characterize latent properties of users or items, such as users' interests, or categories of items. So it is possible to leverage content information to improve personalized ranking. In this paper, we associate contents with implicit feedbacks to develop a non-uniform item sampler for drawing informative training data. For an observed user-item pair, our sampler draws items which belong to the same category with the observed item and are probably viewed by the user. Furthermore, we embed our sampler into BPR framework, and scale BPR to model both implicit feedbacks and contents. Specifically, we propose an comprehensive solution, Content-aware and Adaptive Bayesian Personalized Ranking (CA-BPR) method, which incorporates implicit feedbacks with contents to learn latent vectors for entities as well as content-aware mappings. In a nutshell, our contributions in this paper are listed as follows:

1. We illustrate the reasons about the slow convergence problem in conventional BPR learning.

2. To speed up pairwise learning for personalized ranking, we take the content information into considerations and develop an adaptive sampling strategy to draw informative training data. Our sampling strategy provides a better balance of efficiency and performance than previous strategies, e.g., uniform sampler [4], or adaptive sampler [7].

3. To roundly promote the performance of personalized ranking, we embed our sampler into BPR learning framework, and propose CA-BPR which can learn reliable latent vectors for entities as well as mappings for contents.

4. We conduct a series of experiments to validate proposed methods. The results show that our sampling strategy indeed speed up BPR learning, and CA-BPR can improve personalized ranking by combining contents and implicit feedbacks.

## II. Related work

Factorization methods are popular in personalized recommender systems. They are utilized to deal with various information collected by recommender systems, such as implicit feedbacks [9] [4], attributes of item [5] [2], user profiles [10] and social information [11]. According to the data they studied, these methods can be generally arranged into two branches, i.e., collaborative methods, or content-based methods.

The collaborative methods [4] [12] [8] deal with a mass of users interactions with items, e.g., implicit feedbacks and explicit ratings (they are called as collaborative information). These methods factorize collaborative information, and attempt to learn latent representations for users and items in a shared latent space. Matrix factorization (MF) [13] and its extensions [14] [15] [16] are typical factorization methods to deal with collaborative information. For instance, implicit MF (iMF) [9] extends the basic MF to deal with implicit feedbacks by calculating an adaptive confidence weight for each user-item pair. Although MF extensions can deal with implicit feedbacks, they easily fall into the over-fitting problem because of commonly existing data skew in implicit feedback data sets (the number of positive feedbacks is usually less than one percent of the total number). For alleviating the data skew and learning recommender systems from implicit feedbacks, Bayesian Personalized Ranking (BPR) [4] and its extensions [6] [3] [7] are proposed, which make a pairwise assumption that users are more interested in items that they have selected than the remaining items. Since the pairwise assumption usually derives large scale training data, the learning algorithms are typically based on SGD with uniformly drawn pairs. However, different training samples may have different contributions to the parameter learning, the uniformly sampling strategy usually generates a mass of ineffective training samples and results in slow convergence. In particular, when the number of items is large and the item popularity has long-tail distribution [17], the uniformly sampling strategy will suffer from extremely slow convergence. Thus, to speed up the BPR learning, Rendle et al. further investigate the long-tail effect and utilize long-tail effect to develop non-uniformly item samplers [7]. For a given user, they plan to select items which are popular in a certain domain and have not been selected by the user to consist training pairs. Theoretically, this sampling scheme is very time consuming, because it treats latent factors of items as the popularity indicators of items and needs to reorder items under each domain in each iteration. For amortizing runtime, Rendle et al. have to compromise on predictive performance by reducing times of reordering. On the other hand, aiming to obtain a general scheme to speed up the BPR learning, Zhong et al. [8] try to draw informative training pairs according to preference difference of a user on two unselected items. However, since the number of items in real-world data sets is very large, this strategy has to spend a lot of time on calculating preference differences. Thus, both [8] and [7] are falling into a dilemma in terms of how to balance efficiency and performance, while our adaptive sampling algorithm has a better strategy to cope with this dilemma and has more potential to speed up the BPR learning.

Content-based methods [5] [2] [10] investigate entities in recommender systems from their content information. Generally, this kind of methods utilize contents of entities, such as attributes of item, texts of user, or pixels of picture, to learn latent representations for entities and to alleviate the cold start problem. For example, in Factorization Machines (FM) [2], all kinds of attribute information are concatenated into a feature matrix, and factors associated with attributes are factorized from these content information by rating regression. To alleviate the cold start problem in recommender systems, Map-BPR [5] extends BPR framework to learn content-aware mappings from the space of content information to the latent space. Then, Map-BPR utilize the learned content-aware mappings to learn latent vectors for the new entities which lack enough collaborative information. However, Map-BPR segments the latent vectors learning into two independent parts. This limitation causes that the latent vectors of entities which occur in implicit feedback data set, only indicate collaborative properties of entities but not any content properties. For obtaining more reliable latent vectors, CA-BPR method learns the latent vectors for entities observed in data set from both collaborative information and content information by a unified learning process.

## III. Pairwise preference learning from Implicit Feedback

In this section we first briefly review the BPR algorithm, and then discuss its limitations, i.e., slow convergence and cold start problem.

BPR [4] is a popular personalized ranking framework for dealing with implicit feedback. If the user $u_m$ has selected the item $v_i$ but not selected the item $v_j$, then BPR assumes that, $u_m$ prefers $v_i$ over $v_j$, and defines the pairwise preference of $u_m$ as

$$p\,(i \succ_m j) := \Phi\,(\,x_{mij}\,), \tag{1}$$

where $\Phi\,(x) = 1/(1 + exp\,(-x))$ and $x_{mij} := s\,(u_m, v_i) - s\,(u_m, v_j)$. $s\,(\cdot,\cdot)$ can be any kind of scoring functions which indicates the relevance between the user and the item.

Based on the pairwise preference assumption, the set of all pairwise preference $D_S := \{\,(m,i,j)\,|\,v_i \in I^+_{u_m} \wedge v_j \in I \setminus I^+_{u_m}\,\}$ can be created from implicit feedback data set, where $I^+_{u_m}$ denotes the set of observed items which are selected by the user $u_m$, and a triple $t = (m,i,j)$ represents the user $u_m$ is relevant to the item $v_i$ but irrelevant to the item $v_j$. For simplicity, we call $v_i$ as a positive item of $u_m$, while $v_j$ is a negative one. Given a set of pairwise preference $D_S$, the goal of BPR is to maximize the likelihood of all pairwise preference:

$$\arg\max_{\Theta} \prod_{(m,i,j)\in D_S} p\,(i \succ_m j), \tag{2}$$

which is equivalent to minimize the negative log likelihood:

$$L_{feedback} = -\sum_{(m,i,j)\in D_s} \ln \Phi\,(\,x_{mij}\,) + \lambda||\Theta||^2, \tag{3}$$

where $\Theta$ denotes the set of all latent vectors and $\lambda$ is a hyper-parameter. Since the size of $D_S$ is very large, the learning algorithms of BPR are generally based on SGD with uniformly drawn training triples.

However, the above learning schema often results in slow convergence, because the uniformly sampling draws amount of training pairs which have low contributions on the gradients. More specifically, for a given training sample $(m,i,j) \in D_S$, the stochastic gradient of an arbitrary parameter $\theta \in \Theta$ is:

$$\frac{\partial L_{feedback}}{\partial \theta} = (1 - \Phi\,(\,x_{mij})) \frac{\partial\,(x_{mij})}{\partial \theta}. \tag{4}$$

According to Eq. (4), if $\Phi\,(\,x_{mij}) \rightarrow +1$, the stochastic gradient will approximate to 0, and the contribution of training sample $(m,i,j)$ on approaching the optimization objective is very small. Associating Eq. (4) with Eq. (1), for an observed user-item pair, a negative item should be comparable to the observed item, i.e., the preference scores of a negative item and the observed item, which are provided by the observed user, should be close to each other, otherwise the training sample is inefficient to SGD. Empirically, each user only has viewed a small subset of items, and has provided feedbacks on some of items in this subset. The uniform sampler equally draws negative items from the total set of items. For an observed user-item pair, most uniformly sampled items are not comparable to the observed item or impossible to be viewed by corresponding user. For example, iPhone with toothbrush or iPhone with a non-mainstream cell phone often be sampled by a uniform sampler. Since these training pairs have tiny effect on SGD, the training process often is slow convergence.

Besides, similar to classical factorization techniques, if a user or item lacks enough feedbacks, its latent representation can not be well learned. In real-world data sets, the distributions of user activity and item popularity tend to be long tailed. This causes that most users and items only have a handful feedbacks. Moreover, in real-world recommender systems, new entities may be added into systems at any time. As a result, the BPR framework easily suffers from a cold start problem.

Since the content information of entities can characterize latent properties of users and items. In the following, we first develop a non-uniformly sampling strategy for drawing informative negative items, and then provide a comprehensive solution for obtaining better personalized ranking performance.

## IV. Adaptive sampling strategy

In this section, we combine contents and implicit feedbacks to develop a non-uniform item sampler. Our strategy automatically approximates realistic distribution of data set and adaptively draws informative training pairs.

### A. Overview of Adaptive Sampling Strategy

In most of real-world applications, users assess the items in the same category, and then make their options. Therefore, we hold a self-evident rule, i.e., "apples to apples" to sample informative items for each observed user-item pair. We tend to sample those negative items which are comparable to the observed items as well as have high probability to be viewed by related users. Specifically, for an observed user-item pair $(u_m, v_i)$, we conduct following steps to select a reasonable item $v_j$ as a negative item:

1. According to the categorical distributions of user $u_m$ and item $v_i$, we first infer the event that user $u_m$ selected item $v_i$ happens on which category.

2. Given a category, we further select an item $v_j$ as a negative item, which has a high probability to be browsed by user $u_m$ under the category.

### B. Categorical Distribution of Entity

In many applications, due to the lack of category information, recommender systems have difficulty in accurately providing the categories of users and items. For dealing with this issue, we utilize the latent representation of an entity to approximatively indicate its categories.

Formally, we assume that an entity can belong to multiple categories $C = \{c_1, c_2, ...., c_k\}$, and its categorical distribution follows a power law [7]. Let $y^e_i \in \mathbb{R}^k$ denote the latent vector of entity $e_i$ and the matrix $Y^e = [\,y^e_1, y^e_2, y^e_3, ...\,]$ is latent representation of entities which is learned from both content information and implicit feedback. We consider the classical scenario of recommendation, which has two types of entities, i.e., users (e.g., customers) and items (e.g., movies, books and songs). For clearly, we use a superscript $u$ or $v$ to denote a variable associating with users or items. For example, $y^u_m$ denotes the latent vector of user $u_m$ and $Y^u$ denotes the latent representation matrix of user. To bridge the categorical distribution and the latent vector of entity, we treat the probability that the entity $e_i$ belongs to the category $c \in C$, as a mixture over standardized factors, and define it as

$$p(c\,|\,e_i) \propto exp\,(\,\frac{y^e_{i,c} - \mu_c}{\sigma_c}\,), \tag{5}$$

where $\mu_c = E\,(\,y^e_{*,c}\,)$ and $\sigma_c = Var\,(\,y^e_{*,c}\,)$ denote the empirical mean and variance over all entity factors, respectively. Assuming that the categorical distributions of users and items

are independent, we further derive the probability of observed user-item pair $(u_m, v_i)$ associating with the category $c$ to be a joint probability:

$$p(c \mid u_m, v_i) = p(c \mid u_m)\, p(c \mid v_i). \tag{6}$$

According to the joint probability, we can sample a category $c$ for the observed event that user $u_m$ picked out item $v_i$.

### C. Rank-Invariant of Item List

Under a given category $c$, we aim to draw an item $v_j$, which has high potential to have been browsed by user $u_m$. Intuitively, we can treat the probability $p(c \mid e_i)$ as the ranking score of $e_i$ under the category $c$, and directly draw items according to their ranking scores. However, there exists a gap between browsing probabilities and ranking scores. In real-world applications, items are presented in ranking lists, and the top-ranked items usually have significantly high probabilities to be browsed by users than other items. For instance, the top three items in the ranking list have much higher probabilities to be browsed by users than remaining ones, while the ranking scores of items may only have tiny differences. For bridging this gap, we segment the sampling under a given category into two steps. Specifically, we first sample a ranking place $r$ for candidates from an empirical distribution. Then, we sort items under this category, and return the item at the ranking place $r$ to be a negative item for the observed user-item pair.

Typically, the empirical distribution approximately follows an analytical law, e.g. a Geometric [18] or Zipf [19] distribution. Here, we adopt Geometric distribution to draw the item $v_j$ from the ranking list of the category $c$ :

$$p(v_j \mid c) \propto exp(-r(j)/\lambda) \quad \lambda \in \mathbb{R}^+, \tag{7}$$

where $r(j)$ denotes the ranking place of the item $v_j$, and $\lambda$ is a hyper-parameter which tunes the probability density.

### D. Discussion of Sorting Items

After obtaining the ranking place of negative item, we need to arrange an item into this place. For this purpose, a simple method in [7], which treats latent factors of items as ranking scores and sorts the items according to their ranking scores, can be implemented. However, this method has to reorder items under each category in each iteration, because the latent vectors of items are updated iteratively. This leads to a high computational complexity, because this simple method at least requires a runtime of $O(k\, n \log n)$ in each iteration for reordering, where $n$ is the number of items. For amortizing the runtime, a compromised strategy is also proposed in [7], which reorders items under each category every $n \log n$ iterations. However, this compromise easily leads to local convergence and a degeneration which is randomly sampling items from a randomly selected subset of items. More specifically, since the number of items usually is very large, the ranking lists retain unchange in many iterations. Thus, the sampler in fact degenerates into a poor one which samples items from a given subset of items. It is means that changes continuously happen on a few latent vectors and the learning process easily falls into local convergence. Moreover, since the latent vectors of items are randomly initialized, ranking lists are equal to random sequences until the first reordering happens. If the random ranking lists are not timely updated, the sampler will degenerate into a random sampler which samples item from an unchange subset of items. Therefore, we need a new method to balance efficiency and performance.

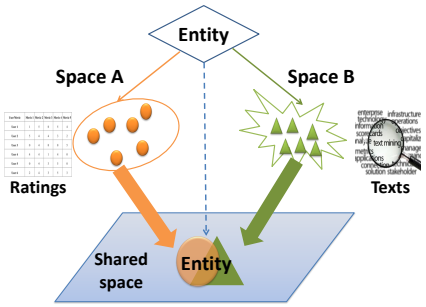


Fig. 1. The schematic of projecting an item from different modalities to a shared latent space. We assume that collaborative information, e.g., ratings, and content information, e.g., text, belong to two different modalities, i.e., space A and space B.

### E. Sampling Algorithm

According to the studies of subspace learning [20] [21], if we project an entity from different modalities to a shared subspace, then its representations in the shared subspace should have some relevances, e.g., complementary or similar. As shown in Figure 1, if we independently project an item from content space and collaborative space to a shared latent space, then we can obtain two latent representations in the shared latent space for the item. We are supposed these two latent representations to be similar or complementary. Inspired by the studies of subspace learning, we plan to avoid the degeneration of randomly sampling from a randomly selected subset of items by initializing the ranking lists according to content information of items. Specifically, we first learn approximate latent representations of items from their content information by feature learning methods, e.g., Conventional Neural Networks (CNN) [22] for images, Latent Dirichlet Allocation (LDA) [23] for texts. Then, we view the approximate latent factors as the ranking scores of items under categories, and sort items under each category. Finally, we initialize ranking lists of items with the sorting results.

Besides, to avoid the problem of local extremum and balance efficiency and performance, we will reorder items under a popular category, if their ranking scores happen significant changes. More specifically, since we can arrange user-item pairs into categories according to Eq. (6), we further count the number of observed user-item pairs under each category, and define a variable $\rho \in \mathbb{R}^k$ to indicate popularity of categories. In each iteration, we first sample a popular category $c$ according to its popularity:

$$p(c \mid \rho) \propto exp\left(\frac{\rho_c - \mu}{\sigma}\right), \tag{8}$$

where $\mu$ and $\sigma$ denote the empirical mean and variance over the variable $\rho$, respectively. Then, we treat current latent factors of items as new ranking scores of items, and measure whether new score vector under category $c$ has significant changes comparing with the old one by a similarity function $sim(\cdot, \cdot)$. If the change of ranking score vector is over given threshold $\delta$, we update ranking scores under category $c$ by $y^v_{*,c}$ which is $c$-th column of latent representation matrix of items, and reorder items under this category.

In summary, we propose an adaptive sampling strategy in Algorithm 1 which selects a negative item for an observed user-item pair with the rule that "apples to apples", i.e., we select negative items which are comparable to their related observed items and have high probability to be viewed by related users. In Algorithm 1, $index\,(c, r)$ returns the item ranked at the place $r$ from the ranking list $l_c \in L$. $x_c \in X$ is the ranking score vector under the category $c$, which is initialized by approximate latent representations of items learned from content information of items. Notice that, in the total learning process, our method usually only needs to reorder items under some categories several times. Our sampling strategy greatly reduces the computational complexity as well as avoid the problem of local extremum.

**Algorithm 1** Content-aware and Adaptive sampling

**Input:**
The observed user-item pair set $S$;
The counters of category popularity $\rho$;
The latent representation matrixes $Y^u$ and $Y^v$;
The ranking scores of items $X = \{x_1, x_2, ...., x_k\}$;
The orders of items $L = \{l_1, l_2, ...., l_k\}$;
**Output:**
The training triple $(u_m, v_i, v_j)$;
The category popularity $\rho$;
Draw a category $c$ from $p\,(c \,|\, \rho)$;
**if** $sim\,\left(x_c, y^v_{*,c}\right) < \delta$ **then**
  Update $x_c$ by $y^v_{*,c}$;
  Reorder items under $c$ and update $l_c$;
**end if**
Draw $(u_m, v_i) \in S$ uniformly;
Draw a category $c$ from $p\,(c \,|\, u_m, v_i)$, $(1 \leqslant c \leqslant k)$;
$\rho_c + +$;
Draw a rank $r$ from $p\,(r) \propto exp\,(-r\,/\,\lambda\,)$, $(1 \leqslant r \leqslant n)$;
$v_j \leftarrow \begin{cases} index\,(c, r) & if\ \ sgn\,(\,y^u_{m,c}\,) = 1, \\ index\,(c, n - r - 1) & else \end{cases}$

## V. Content-aware and Adaptive BPR

In the above sections, we have illustrated how to improve the BPR learning by an adaptive item sampler, and learn the latent factors of entities only considering implicit feedbacks. However, in real-world recommender systems, entities do not always have enough collaborative information, e.g., new entities may be added into the systems at any time. Therefore, we propose a comprehensive solution for personalized ranking, Content-aware and Adaptive Bayesian Personalized Ranking (CA-BPR), which is based on our adaptive sampling strategy and incorporates implicit feedbacks and contents into a unified personalized ranking framework. CA-BPR can learn reliable latent vectors for entities and mappings for contents.

### A. Learning Content-Aware Mappings

Here, we formally present an unsupervised solution for learning the content-aware mappings. We first use the matrix $A^e = [\,a^e_1, a^e_2, a^e_3, ...\,]$ to denote the content features of entities. Then, we present the objective function to learn the content-aware mappings:

$$L_{content} = ||A^e W^e - Y^e||^2_F, \tag{9}$$

where $W^e \in \mathbb{R}^{d^e \times k}$ denotes a mapping matrix, and $k$ is the dimension of latent vectors.

### B. Parameter Inference of CA-BPR

Generally speaking, the optimization problem expressed by Eq. (10) has infinite solutions because of lacking supervised information. Fortunately, according to the studies of subspace learning, we can learn a latent matrix $\widetilde{Y^e}$ from implicit feedbacks, and use $\widetilde{Y^e}$ to be an approximation of $Y^e$. As a result, we treat $\widetilde{Y^e}$ as pseudo labels, and substitute it into Eq. (9), then we rewrite the objective function as

$$L_{content} = ||A^e W^e - \widetilde{Y^e}||^2_F. \tag{10}$$

With the pseudo labels, we not only can optimize our objective function, but also can jointly learn the mapping matrix $W^{(e)}$ from both collaborative information and content information. Therefore, the overall objective function of CA-BPR with latent vectors and content-aware mappings is expressed as

$$\begin{aligned} &\arg\min_{\Theta, W} \ \ L_{feedback} + L_{content} = \\ &- \sum_{(m,i,j) \in D_s} \ln \Phi\left(\,(y^v_i - y^v_j)\, y^u_m\right) + \lambda ||\theta||^2 \\ &+ \frac{1}{2} \sum_{e \in \{u,v\}} \lambda^e ||W^e||^2_F + ||A^e W^e - Y^e||^2_F. \end{aligned} \tag{11}$$

For learning the parameters $Y^u$, $Y^v$, $W^u$ and $W^v$ in Eq. (11), we develop an alternative optimization algorithm to learn latent vectors and mapping matrices iteratively. Specifically, our algorithm uses SGD with our adaptive sampling strategy to learn the latent vectors and implements matrix decomposition with the alternative least squares to learn the mapping matrices.

In each iteration, when we are updating the latent factor matrix $Y^e$, we set the mapping matrix $W^e$ to be a constant and treat $L_{content}$, the entire optimization objective of content information, to be a regularizer. Thus, the gradient of an arbitrary latent parameter $\theta$ is

$$\begin{aligned} \frac{\partial L}{\partial \theta} = &\sum_{(m,i,j) \in D_s} \left(1 - \Phi\left(\,(y^v_i - y^v_j)\, y^u_m\right)\right) \frac{\partial\,((y^v_i - y^v_j)\, y^u_m)}{\partial \theta} \\ &+ \frac{\partial \sum_{e \in \{u,v\}} \lambda^e \left(||A^e W^e - Y^e||^2_F\right)}{\partial\, \theta} + \lambda\, \theta. \end{aligned} \tag{12}$$

The updating rule for parameter $\theta$ is $\theta = \theta + \eta\, \frac{\partial L}{\partial \theta}$, where $\eta$ is the learning rate. On the other hand, given a latent factor matrix $Y^e$, we view $Y^e$ as pseudo labels and treat $L_{feedback}$ as a constant. Thus, the derivative of objective is

$$\frac{\partial L}{\partial W^e} = (A^e)^T (A^e W^e - Y^e) + \lambda^e\, W^e. \tag{13}$$

Let $\frac{\partial L}{\partial W^e} = 0$, the updating rule for $W^{(e)}$ can be derived as

$$W^e = (\,(A^e)^T A^e + \lambda^e \mathbb{E}\,)\, A^e Y^e, \tag{14}$$

where $\mathbb{E} \in \mathbb{R}^{k \times k}$ is an identity matrix.

In a nutshell, the algorithm for learning parameters of CA-BPR is summarized in Algorithm 2. It mainly repeats two learning steps until the parameters reach convergence, i.e., it first learns the latent factors of entities from implicit feedbacks, by SGD with our adaptive item sampler, and then, it learns the content-aware mapping matrices by matrix decomposition.

**Algorithm 2** Learn parameters for CA-BPR

**Input:**
The observed user-item pair set $S$;
The feature matrix of items $F$;
The content feature of entities $A := \{A^u, A^v\}$
**Output:**
$\Theta := \{Y^u, Y^v\}$, $W := \{W^u, W^v\}$;

```
Initialize Θ and W with uniform(−√6/k, √6/k);
Standardized Θ;
Initialize the popularity of categories ρ randomly;
repeat
    Draw a triple (m, i, j) with Algorithm 1;
    for each latent vector θ ∈ Θ do
        θ ← θ + η ∂L/∂θ;
    end for
    for each W^e ∈ W do
        Update W^e with the rule defined in Eq. (14);
    end for
until convergence
```

## VI. Experiment

In this section, comparing with the state-of-the-art methods, we conduct experiments to validate our sampling strategy and CA-BPR method on two public data sets. First we investigate convergence rate of our sampling strategy. Then, we evaluate ranking quality of CA-BPR by Area Under the ROC Curve (AUC) and Mean Average Precision (MAP). Finally, on cold start problem, we study the performance of compared methods by simulating new items recommendation.

### A. Data sets and Tasks

We use two real-world data sets for experiments, i.e., DBLP[1] and MovieLens[2], and carry out training and testing on randomly split training (80%) and testing (20%) data.

**DBLP** contains 2,084,055 papers with 2,244,018 citations. Each paper may be associated with abstract, authors, published year, and title. We preprocess the DBLP data to be an experimental data set in a similar way as [7]. More specifically, we sample 1,000 authors who published no more than 5 papers and cited 5-100 papers. Thus, we obtain 1,000 authors, 16,313 papers and 23,506 author-paper pairs. Each author-paper pair denotes a relation of the author cited the paper. In experiments, we treat texts in published papers of an author as the content information of this author, and paper text as content information of the paper. We use the term-frequency over texts as features of content information. Our task is to predict the personalized ranking of citing papers for each author.

**MovieLens** includes 100,000 ratings with 943 users and 1682 movies. Each user has rated at least 20 movies. In experiments, the occupational description of a user is treated as content information of the user, and the key words in the title of a movie are viewed as content information of the movie. Using the same processing method in [24], we do not use rating values but just binary rating events by assuming that users tend to rate watched movies. Thus, for a specific user, our task is to predict the potential ranking list of movies.

Comparing these two data sets, we observe that:

- DBLP data set has a heavier long-tailed effect than the MovieLens data set. And the implicit rating matrix of DBLP data set is sparser than that of MovieLens.
- Each entity in MovieLens has its corresponding content information, but entities in DBLP always lack corresponding texts. In DBLP, many authors have incomplete content information and some papers do not have content information.

[1]http://arnetminer.org/citation

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

TABLE I. Characteristics of Compared Methods

| Method | Content | Feedback | Sampling |
|---|---|---|---|
| MF | no | implicit | – |
| FM | yes | implicit/explicit | – |
| BPR-MF | no | implicit | uniform |
| Map-BPR | yes | implicit | uniform |
| Ada-BPR | no | implicit | non-uniform |
| **CA-BPR** | **yes** | **implicit** | **non-uniform** |

### B. Compared Methods

To investigate our sampling strategy and the CA-BPR method, we compare our methods with state-of-the-art methods outlined below.

- **MF** [25] is a classical factorization method, which is commonly used to deal with collaborative information.
- **BPR-MF** [4] is a matrix factorization method derived under the BPR learning framework. This method utilizes uniformly sampling strategy to obtain training samples.
- **Ada-BPR** [7] is a recent work aiming at speeding up BPR learning with a non-uniformly sampling strategy.
- **FM** [2] is a general framework, which can be used to deal with both content information and collaborative information. To evaluate the effect of content information, here we use FM to only deal with contents.
- **Map-BPR** [5] is a work for alleviating cold start problem in BPR, which independently deals with implicit feedbacks and contents.

Table I summarizes the characteristics of these compared methods. In addition, we view user-item pairs which do not occur in implicit feedback data set as negative feedbacks. Since the data sets we used existing data skew, i.e., the number of positive feedbacks is far less than the number of negative feedbacks, the training of FM and MF is easy to suffer from over-fitting. To avoiding the over-fitting caused by data skew, we train FM and MF with the training data sets which contain randomly drawn negative feedbacks, i.e., the proportions of negative feedbacks and positive feedbacks are $50:1$ on DBLP and $100:1$ on MovieLens.

### C. Results and Discussion

*1) Convergence:* Figure 2 shows the convergence comparisons on DBLP and MovieLens, where the performance of methods is evaluated by AUC and likelihood of objective in each training epoch. The hyper-parameters of compared methods, i.e., learning rate and factor dimensionality, are set to be the same values in experiments. Figure shows that the methods with non-uniformly sampling strategies usually can achieve faster convergence rates than that of BPR-MF. Moreover, the method with our sampling strategy not only achieves approximate convergence rates with Ada-BPR, but also reaches

TABLE II. PERFORMANCE COMPARISON ON THE DBLP AND MOVIELENS DATA SETS WITH THREE DIFFERENT FACTOR DIMENSIONALITIES( K = 10; 20; 50). THE PERFORMANCE OF APPROACHES IS EVALUATED BY MAP AND AUC.

| Method | DBLP | | | | | | Movielens | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MAP | | | AUC | | | MAP | | | AUC | | |
| | K=10 | K=20 | K=50 | K=10 | K=20 | K=50 | K=10 | K=20 | K=50 | K=10 | K=20 | K=50 |
| MF | 0.096 | 0.091 | 0.093 | 0.611 | 0.606 | 0.616 | 0.235 | 0.238 | 0.240 | 0.794 | 0.793 | 0.791 |
| FM | 0.066 | 0.067 | 0.069 | 0.611 | 0.605 | 0.621 | 0.144 | 0.150 | 0.147 | 0.657 | 0.658 | 0.659 |
| BPR-MF | 0.226 | 0.227 | 0.257 | 0.611 | 0.606 | 0.616 | 0.305 | 0.288 | 0.271 | 0.850 | 0.843 | 0.830 |
| Ada-BPR | 0.244 | 0.183 | 0.228 | 0.624 | 0.579 | 0.582 | 0.298 | 0.267 | 0.237 | 0.825 | 0.821 | 0.771 |
| Map-BPR | 0.222 | 0.247 | 0.278 | 0.612 | 0.617 | 0.616 | 0.306 | 0.291 | 0.267 | 0.850 | 0.842 | 0.829 |
| **CA-BPR** | **0.279** | **0.276** | **0.297** | **0.660** | **0.652** | **0.651** | **0.320** | **0.316** | **0.307** | **0.852** | **0.850** | **0.847** |

TABLE III. TOP-N RECOMMENDATION EVALUATED BY MAP@N (N=3, 10, 20) AND RATING PREDICTION EVALUATED BY RMSE, ON DBLP (K=50) AND MOVIELENS (K=50).

| Method | DBLP | | | | MovieLens | | | |
|---|---|---|---|---|---|---|---|---|
| | MAP@3 | MAP@10 | MAP@20 | RMSE | MAP@3 | MAP@10 | MAP@20 | RMSE |
| MF | 0.063 | 0.221 | 0.207 | 0.312 | 0.346 | 0.385 | 0.344 | 0.431 |
| FM | 0.025 | 0.095 | 0.100 | 0.709 | 0.168 | 0.221 | 0.207 | 0.495 |
| BPR-MF | 0.215 | 0.257 | 0.260 | 0.922 | 0.322 | 0.376 | 0.329 | 1.411 |
| Ada-BPR | 0.220 | 0.261 | 0.263 | 0.940 | 0.212 | 0.271 | 0.245 | 3.060 |
| Map-BPR | 0.207 | 0.245 | 0.243 | 1.027 | 0.275 | 0.325 | 0.293 | 3.987 |
| **CA-BPR** | **0.246** | **0.283** | **0.282** | **0.667** | **0.367** | **0.398** | **0.357** | **1.375** |

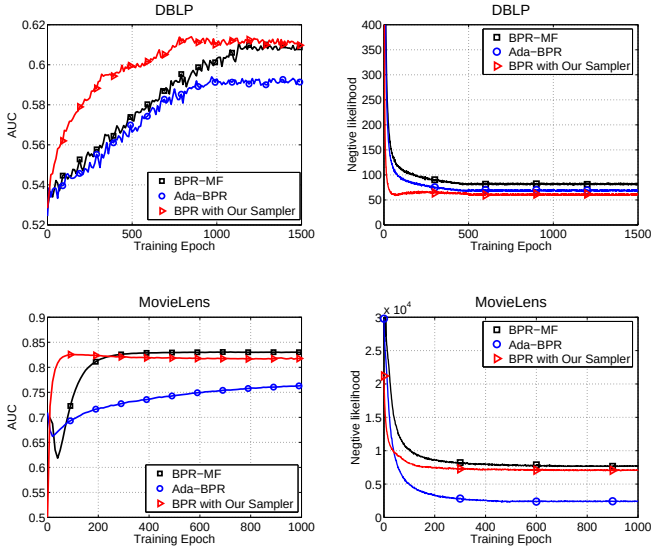


Fig. 2. Convergence comparison on DBLP and MovieLens. Each approach is trained by given the same parameters, i.e., the learning rate is 0.15 on DBLP and 0.012 on MovieLens; the factor dimensionality is 50 on both data sets; the training epoch is 1500 on DBLP and 1000 on MovieLens.

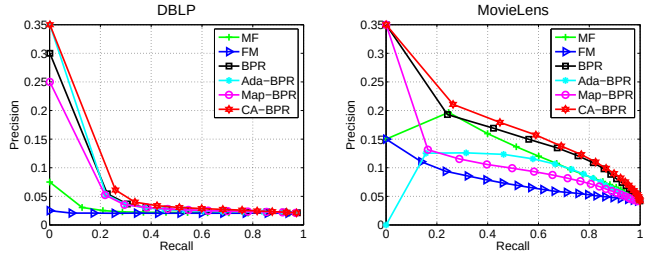


Fig. 3. Precision-Recall curves on DBLP (K=50) and MovieLens (K=50).

TABLE IV. THE PERFORMANCE OF NEW ITEMS RECOMMENDATION ON THE DBLP DATA SET. THE PERFORMANCE IS MEASURED BY MAP@N (N=3, 10, 20). WE ONLY EVALUATE THE USER-ITEM PAIRS WHEN THE ITEMS ARE NOT APPEARED IN THE TRAINING SET.

| Method | MAP@3 | MAP@10 | MAP@20 |
|---|---|---|---|
| MF | 0.000 | 0.000 | 0.001 |
| FM | 0.016 | 0.034 | 0.043 |
| BPR-MF | 0.037 | 0.057 | 0.075 |
| Ada-BPR | 0.038 | 0.053 | 0.067 |
| Map-BPR | 0.050 | 0.080 | 0.087 |
| **CA-BPR** | **0.057** | **0.085** | **0.098** |

better convergence points than Ada-BPR in terms of AUC. This result indicates our sampling strategy can obtain a better balance of efficiency and performance, and has more potential to speed up the BPR learning. In addition, although Ada-BPR has competitive convergence rates in terms of convergence on negative likelihood or AUC, its performance, in terms of convergence point of AUC, is worst among of compared methods. One possible reason is that Ada-BPR reorders items under each category after $nlogn$ iterations for amortizing runtime. This compromise causes that Ada-BPR continuously samples items from an unchanged subset of items in many iterations, and easily falls into local optimal.

*2) Ranking Quality:* Table II gives the ranking performance with different factor dimensionalities. Since content information may be noisy and incomplete, FM only is used to deal with content information, usually can not get reliable ranking performance. On the other hand, due to the lack of enough collaborative information, the methods only considering implicit feedbacks, e.g., MF, BPR-MF and Ada-BPR, also can not get ideal results. Owing much to combining implicit feedbacks with contents, CA-BPR consistently outperforms other methods in terms of MAP and AUC. Besides, we note that, the performance of Ada-BPR often strikingly fluctuate with different dimensionalities, e.g., DBPL (K=20) and MovieLens (K=50). This is another evident that Ada-BPR easily falls into local optimal. Meanwhile, since CA-BPR is developed on our adaptive sampling strategy, it can avoid local optimal problem.

*3) Recommendation Performance & Rating Prediction:* Figure 3 shows the performance of different methods with varying number of recommendations. From that we can see, due to a comprehensive solution for dealing with content information and collaborative information, CA-BPR consistently outperforms other methods. Besides, Ada-BPR has better performance on DBLP than on MovieLens. This is because the sampling strategy of Ada-BPR applys long-tail effect to speed up convergence, and DBLP data set exists a heavier long-tail effect than MovieLens data set. This result may indicate that

Ada-BPR may be hard to obtain reliable performance, when the data sets do not have significant long-tail effect.

We further evaluate the ability of compared methods in the tasks of Top-N recommendation and rating prediction. Table III shows the results in terms of MAP@N and RMSE. Owing much to sampling informative training data, CA-BPR performs better than all baselines in terms of MAP@N. Since users are more likely to care about several items which are on the top places, this result indicates that CA-BPR can well fit requirements of real-world recommender systems. Besides, CA-BPR outperforms other BPR extensions in the task of rating prediction in terms of RMSE. Note that our rating prediction results should not be directly compared to MF or FM because their training environments are different, i.e., MF and FM are trained on training sets which fix the proportions of negative feedbacks and positive feedbacks.

*4) Cold Start Problem:* For investigating the cold start problem, we evaluate the ability of compared methods on the task of new items recommendation. More specifically, we only evaluate the user-item pairs which are occurring in testing set but their items are not occurring in training set. Table IV provides recommendation performance in terms of MAP@N. Due to the lack of collaborative information, the methods without considering contents usually have worse performance than content-aware methods. Moreover, CA-BPR can obtain the best results among of content-aware methods. Besides, we can observe that, FM only outperforms MF, and all BPR extensions, whether they take contents into account or not, can outperform FM. There may be two reasons about this result. Firstly, many new items in DBLP data set also lack corresponding content information. FM only can provide random predictive results on this part of new items. Secondly, since content information usually contains noisy, the performance of FM may degenerate.

## VII. Conclusions

Traditional pairwise learning algorithms for personalized ranking usually have two fundamental limitations, i.e., long training process caused by uniformly sampling strategy and neglect of content information of entities. In this work, we have illustrated our solutions for dealing with these limitations. Specifically, to speed up BPR learning, we have developed an adaptive sampling strategy, which utilizes content information to obtain a better balance of efficiency and performance. To alleviate cold start problem and further improve performance, we have proposed CA-BPR, which embeds our sampling strategy to efficiently learn parameters. CA-BPR can combine implicit feedbacks and contents to learn reliable latent factors for entities as well as content-aware mappings. Comprehensive experiments have validated our solutions by comparing with the state-of-the-art methods. However, in real-world applications, content information usually includes multiple types, noisy and redundant, which may lead to performance degeneration of our solutions. In the future, for obtaining more reliable performance in such scenarios, we plan to devise a more robust solution based on this work.

## References


[1] Yue Shi, Martha Larson, and Alan Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering. In *RecSys*, pages 269–272, 2010.

[2] Steffen Rendle. Factorization machines with libFM. *TIST*, 3(3):57:1–57:22, 2012.

[3] Shuang Qiu, Jian Cheng, Ting Yuan, Cong Leng, and Hanqing Lu. Item group based pairwise preference learning for personalized ranking. In *SIGIR*, pages 1219–1222, 2014.

[4] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[5] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, Steffen Rendle, and Lars Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *ICDM*, pages 176–185, 2010.

[6] Weike Pan and Li Chen. Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering. In *IJCAI*, pages 2691–2697, 2013.

[7] Steffen Rendle and Christoph Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *WSDM*, pages 273–282, 2014.

[8] Hao Zhong, Weike Pan, Congfu Xu, Zhi Yin, and Zhong Ming. Adaptive pairwise preference learning for collaborative recommendation with implicit feedbacks. In *CIKM*, pages 1999–2002, 2014.

[9] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272, 2008.

[10] Liangjie Hong, Aziz S Doumith, and Brian D Davison. Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In *WSDM*, pages 557–566, 2013.

[11] Hao Ma, Dengyong Zhou, Chao Liu, and Lyu. Recommender systems with social regularization. In *ICDM*, pages 287–296, 2011.

[12] Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. Recommendation in heterogeneous information networks with implicit user feedback. In *RecSys*, pages 347–350, 2013.

[13] Sheetal Girase, Debajyoti Mukhopadhyay, et al. Role of matrix factorization model in collaborative filtering algorithm: A survey. *arXiv preprint arXiv:1503.07475*, 2015.

[14] Yanen Li, Jia Hu, ChengXiang Zhai, and Ye Chen. Improving one-class collaborative filtering by incorporating rich user information. In *CIKM*, pages 959–968, 2010.

[15] Rainer Gemulla, Erik Nijkamp, Peter J Haas, and Yannis Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *SIGKDD*, pages 69–77, 2011.

[16] Wancai Zhang, Hailong Sun, Xudong Liu, and Xiaohui Guo. Temporal qos-aware web service recommendation via non-negative tensor factorization. In *WWW*, pages 585–596, 2014.

[17] Anja Feldmann and Ward Whitt. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. In *INFOCOM*, volume 3, pages 1096–1104, 1997.

[18] Kuansan Wang, Toby Walker, and Zijian Zheng. Pskip: estimating relevance ranking quality from web search clickthrough data. In *SIGKDD*, pages 1355–1364, 2009.

[19] Lada A Adamic and Bernardo A Huberman. Zipfs law and the internet. *Glottometrics*, 3(1):143–150, 2002.

[20] Raghavendra Udupa and Mitesh Khapra. Improving the multilingual user experience of wikipedia using cross-language name search. In *NACACL-HLT*, pages 492–500, 2010.

[21] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Coviello, Gabriel Doyle, Gert RG Lanckriet, and Nuno Levy. A new approach to cross-modal multimedia retrieval. In *MM*, pages 251–260, 2010.

[22] Yoshua Bengio. Deep learning of representations: Looking forward. In *Statistical language and speech processing*, pages 1–37. 2013.

[23] Andrew Y. Ng Michael I. Jordan, David M. Blei. Latent dirichlet allocation. *JMLR*, 2003.

[24] Asela Gunawardana and Christopher Meek. A unified approach to building hybrid recommender systems. In *RecSys*, pages 117–124, 2009.

[25] Wei-Sheng et al. A learning-rate schedule for stochastic gradient methods to matrix factorization. In *PAKDD*, pages 442–455. 2015.