Journal of the American Statistical Association

ISSN: 0162-1459 (Print) 1537-274X (Online) Journal homepage: https://www.tandfonline.com/loi/uasa20

Scalable Collaborative Ranking for Personalized Prediction

Ben Dai, Xiaotong Shen, Junhui Wang & Annie Qu

To cite this article: Ben Dai, Xiaotong Shen, Junhui Wang & Annie Qu (2020): Scalable Collaborative Ranking for Personalized Prediction, Journal of the American Statistical Association, DOI: 10.1080/01621459.2019.1691562

To link to this article: https://doi.org/10.1080/01621459.2019.1691562

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION
2019, VOL. 00, NO. 0, 1–9: Theory and Methods
https://doi.org/10.1080/01621459.2019.1691562

Scalable Collaborative Ranking for Personalized Prediction

Ben Dai[a,b], Xiaotong Shen[a], Junhui Wang[b], and Annie Qu[c]

[a]School of Statistics, University of Minnesota, Minneapolis, MN; [b]School of Data Science, City University of Hong Kong, Kowloon Tong, Hong Kong; [c]Department of Statistics, University of Illinois at Urbana Champaign, Champaign, IL

ABSTRACT

Personalized prediction presents an important yet challenging task, which predicts user-specific preferences on a large number of items given limited information. It is often modeled as certain recommender systems focusing on ordinal or continuous ratings, as in collaborative filtering and content-based filtering. In this article, we propose a new collaborative ranking system to predict most-preferred items for each user given search queries. Particularly, we propose a ψ-ranker based on ranking functions incorporating information on users, items, and search queries through latent factor models. Moreover, we show that the proposed nonconvex surrogate pairwise ψ-loss performs well under four popular bipartite ranking losses, such as the sum loss, pairwise zero-one loss, discounted cumulative gain, and mean average precision. We develop a parallel computing strategy to optimize the intractable loss of two levels of nonconvex components through difference of convex programming and block successive upper-bound minimization. Theoretically, we establish a probabilistic error bound for the ψ-ranker and show that its ranking error has a sharp rate of convergence in the general framework of bipartite ranking, even when the dimension of the model parameters diverges with the sample size. Consequently, this result also indicates that the ψ-ranker performs better than two major approaches in bipartite ranking: pairwise ranking and scoring. Finally, we demonstrate the utility of the ψ-ranker by comparing it with some strong competitors in the literature through simulated examples as well as Expedia booking data. Supplementary materials for this article are available online.

ARTICLE HISTORY
Received November 2018
Accepted November 2019

KEYWORDS
Bipartite ranking; Discounted cumulative gain; Latent factor models; Matrix factorization; Mean average precision; Recommender systems.

1. Introduction

The rapid growth of e-commerce makes personalized prediction indispensable, as it largely relies on the prediction of each individual customer's preference over a huge collection of products. Personalized prediction has been widely employed in a wide range of applications, including movie recommendations (MovieLens; Miller et al. 2003), restaurant or hotel booking (Entree; Burke 2002), news article suggestions (Daily learner; Billsus and Pazzani 2000), and product marketing and advertisement (Amazon; Linden, Smith, and York 2003). For example, for hotel booking recommendations, a list of most preferred hotels is crucial to enhance a customer's experience of booking.

In the literature, personalized prediction is popularly modeled within the framework of recommender systems, such as collaborative filtering (Paterek 2007; Salakhutdinov, Mnih, and Hinton 2007; Wang, Wang, and Yeung 2015; Mao, Chen, and Wong 2019) and content-based filtering (Lang 1995; Billsus and Pazzani 2000; Middleton, Shadbolt, and De Roure 2004), where ordinal preference ratings are often treated as a continuous variable to facilitate computing. Despite the successes of these approaches, challenges arising from personalized prediction remain persistently unsolved. First, methods designed for continuous or ordinal preference ratings are not appropriate and effective for describing binary preferences in terms of performance and evaluation criteria. Second, personalized prediction is more useful for providing a short list of top-preferred items as opposed to a complete list of items with estimated preference scores. Therefore, ranking has become more relevant than classification or regression in personalized prediction.

In this article, we propose a collaborative ranking method, where each user's preferences over items are binary, such as in hotel booking. Our primary goal is to provide personalized ranking of a short list of most-preferred items, which differs from conventional recommender systems in that our approach focuses on ranking items based on their relevance to observations from users such as queries instead of estimating user-specific preferences. One motivating example is the Expedia data challenge, which aims to recommend a list of most preferred hotels for a searching query. Specifically, a query concerns the event of a customer's searching on hotels, including date, time, searching destination, length of stay, and children counts. In this situation, information regarding search queries and hotels based on observed preferences can be pooled to produce appropriate ranking of potential hotels for future queries. This differs dramatically from conventional ranking in that it considers a common set of users and items, and thus the collaborative information across users and items can greatly enhance ranking accuracy.

It is important to develop an evaluation criterion to measure the ranking accuracy. The losses for bipartite ranking include

CONTACT Annie Qu anniequ@illinois.edu Department of Statistics, University of Illinois at Urbana Champaign, Champaign, IL 61820.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JASA.

the sum loss (Ailon 2014), pairwise zero-one loss (Clémençon, Lugosi, and Vayatis 2008), discounted cumulative gain (DCG; Valizadegan et al. 2009), and mean average precision (MAP; Yue et al. 2007). While the sum loss, DCG, and MAP reward correctly identified top-ranked items more than other correctly ranked items, the pairwise zero-one loss focuses on the preference ordering of each item pair. These loss functions have been adopted for various ranking methods, yet very little is known about the relations among them. As a result, there lacks a benchmark for comparison under different loss functions. Moreover, it is computationally intractable to minimize these loss functions in general, except for the pairwise zero-one loss as it reduces to the pairwise classification loss with feasible computational algorithms (Burges et al. 2005; Clémençon, Lugosi, and Vayatis 2008). In addition, theoretical properties under these loss functions are still unknown. Importantly, many binary ranking systems involve both continuous and discrete predictors, such as the date, time, searching destination, distance, and hotel ID such as in the Expedia dataset. Therefore, it is highly desirable to develop a collaborative ranking method which has a sound theoretical foundation, is computationally scalable, and can incorporate collaborative information from users, items, and search queries to achieve high ranking accuracy.

In this article, we show that the regrets defined by the aforementioned bipartite ranking losses can be upper-bounded by the regret defined by the pairwise zero-one loss. To our knowledge, this is the first work to integrate the regrets of various bipartite ranking losses into one framework. On this ground, we construct a nonconvex surrogate pairwise ψ-loss, which builds up a theoretical foundation for developing accurate collaborative ranking methods. In contrast, as showed in Calauzenes, Usunier, and Gallinari (2012), there is no convex surrogate loss calibrated with the mean average precision loss. Moreover, this unified framework allows us to develop a regularized formulation derived from the surrogate pairwise ψ-loss. In addition, we propose a ranking function motivated by matrix factorization to facilitate user-query-item interaction, which effectively integrates the collaborative information from users, items, and search queries by imposing a nested structure of user-specific queries. In contrast to conventional recommender systems based on estimation of conditional expectation, the ψ-ranker directly targets a bipartite ranking, which not only produces higher accuracy in ranking as measured by the four ranking losses (cf., Lemma 3 and Theorem 1), but also utilizes nested structures to facilitate computation in ranking.

Computationally, we develop a block successive upper-bound minimization (BSUM) scheme together with difference of convex (DC) programming to solve the optimization with two-levels of nonconvex components corresponding to the ψ-loss and matrix factorization. Theoretically, we establish a probabilistic error bound of the pairwise-regret of the proposed ψ-ranker, and derive a sharp rate of convergence. The established rate of convergence also shows that the ψ-ranker performs better than the pairwise ranking and scoring methods under any of the aforementioned four losses. As a by-product, our result also indicates that the ψ-ranker unifies the error rate of the pairwise ranking and scoring approaches in the bipartite ranking. In addition, the ψ-ranker based on a scoring approach reduces computational complexity while achieving desirable theoretical properties. In contrast, other surrogate loss functions, such as exponential or logistic losses (Kotlowski, Dembczynski, and Huellermeier 2011; Agarwal 2014), have a slower rate of convergence or could be inconsistent, for example, the hinge loss. Finally, the proposed method is employed to analyze the Expedia dataset, containing over 550,000 browsing records from over 200,000 searching queries and 60,000 hotels.

The rest of the article is organized as follows. Section 2 introduces commonly used loss functions in bipartite ranking, where a Bayes ranker and the relations among the regrets of various loss functions are established. In addition, we propose a regularized ψ-ranker based on a surrogate pairwise ψ-loss, incorporating the collaborative information among users, items, and search queries. Section 3 develops a scalable computation method. Section 4 derives finite-sample risk bounds for the proposed method. Section 5 evaluates the proposed method in simulation studies and a real application to the Expedia data challenge. Section 6 provides a brief discussion. The technical proofs are contained in the supplementary materials.

2. Method

In this section, we propose a collaborative ranking method, called ψ-ranker, for bipartite ranking systems. Suppose that the binary preferences of T items responding to N searching queries are in a form $\boldsymbol{R} = (\boldsymbol{R}_{\mathbf{Q}_i})_{i=1}^N$ with $\boldsymbol{R}_{\mathbf{Q}_i} = \left(\mathbf{Q}_i, \mathbf{Z}_{it}, Y_{it}\right)_{t=1}^T$, where $\mathbf{Q}_i = (Q_{i1}, Q_{i2}, \ldots, Q_{iG})'$ is an ith discrete-valued searching query vector with $'$ denoting the transpose, G is the length of the query vector, $\mathbf{Z}_{it} \in \mathbb{R}^p$ is a vector of continuous query-item predictors, and $Y_{it} \in \{0, 1\}$ with 1 indicating the preference of item t for query $\mathbf{Q}_i$ and 0 otherwise. For instance, for the Expedia hotel booking dataset, each query $\mathbf{Q}_i$ is comprised of the ith browsing record, including date, time, and searching destination, $\mathbf{Z}_{it}$ consists of the length of stay, children count, hotel review score, hotel location score, hotel rating, price, and distance between the user's location and a hotel t, and Y_{it} is an indicator of whether a user books the hotel t or not. We assume that $(\boldsymbol{R}_{\mathbf{Q}_i})_{i=1}^N$ are independent, and $(\mathbf{Z}_{it}, Y_{it})_{t=1}^T$ are conditional independent and identically distributed given each $\mathbf{Q}_i$. However, $\left(\mathbf{Q}_i, \mathbf{Z}_{it}, Y_{it}\right)$'s are not independent due to the shared query $\mathbf{Q}_i$ in every batch of T observations. The primary goal is to construct a ranking function $f(\mathbf{Q}_i, \mathbf{Z}_{it})$ based on the collaborative information across queries and items, which provides accurate ranking of items with top preference given a query.

2.1. Ranking Loss Functions

A number of loss functions have been proposed to assess the accuracy of bipartite ranking (Chaudhuri and Tewari 2017), including the sum loss, the pairwise zero-one loss, one minus DCG, and one minus MAP. For the latter two, we denote them as the discounted cumulative error (DCE) and the mean average error (MAE). Specifically, given a sample $\boldsymbol{R}$, these losses on a

given ranking function $f(\mathbf{Q}, \mathbf{Z})$ are defined as

$$\text{Sum-loss:} L_{\text{sum}}(f, \boldsymbol{R}) = \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \frac{|\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}|}{T} Y_{it}; \quad (1)$$

$$\text{Pairwise-loss:} L_{\text{pair}}(f, \boldsymbol{R}) = \frac{1}{NT^2} \sum_{i=1}^{N} \sum_{t,t'} I(Y_{it} > Y_{it'}) I\big(f(\mathbf{Q}_i, \mathbf{Z}_{it}) \le f(\mathbf{Q}_i, \mathbf{Z}_{it'})\big); \quad (2)$$

$$\text{DCE:} L_{\text{dce}}(f, \boldsymbol{R}) = 1 - \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \big(\log_2(1 + |\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}|)\big)^{-1} Y_{it}; \quad (3)$$

$$\text{MAE:} L_{\text{mae}}(f, \boldsymbol{R}) = 1 - \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} |\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}|^{-1} \sum_{t' \in \mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}} Y_{it} Y_{it'}, \quad (4)$$

where $\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})} = \{t' : f(\mathbf{Q}_i, \mathbf{Z}_{it'}) \ge f(\mathbf{Q}_i, \mathbf{Z}_{it})\}$ consists of all items ranked higher than $\mathbf{Z}_{it}$ by f for query $\mathbf{Q}_i$, $|\cdot|$ denotes the set cardinality, and $I(\cdot)$ is an indicator function. To rank, we assume, without loss of generality, that $f(\mathbf{Q}_i, \mathbf{Z}_{it})$ has distinct values at different $\mathbf{Z}_{it}$'s with probability 1. Specifically, the pairwise zero-one loss evaluates the relative order of any two items, while the sum loss, DCE, and MAE are defined based on $|\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}|$ to penalize more when top-ranked items are misidentified.

Note that that (2) slightly differs from the classical pairwise-loss for a continuous response $Y \in \mathbb{R}$ in Clémençon, Lugosi, and Vayatis (2008), which imposes an additional unnecessary penalty to a tie $Y_{it} = Y_{it'}$, that almost never occurs to a continuous response $Y \in \mathbb{R}$, but can not be ignored in the binary case $Y \in \{0, 1\}$. In contrast, (2) overcomes this difficulty for a binary response.

For each loss function L, we define the ideal ranker as $f_L^* = \text{argmin}_f E\big(L(f, \boldsymbol{R})\big)$, where the expectation is taken with respect to $\boldsymbol{R}$ and the minimization is taken over all functions. Lemma 1 shows that all the aforementioned loss functions lead to the same Bayes ranker defined by the relative order of $P(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it})$, where $P(\cdot \mid \mathbf{Q}_i, \mathbf{Z}_{it})$ is the conditional probability of Y_{it} given $\mathbf{Q}_i$ and $\mathbf{Z}_{it}$.

Lemma 1 (Characterization of f_L^*). For any loss function L in (1)–(4), the rank of $(\mathbf{Z}_{i1}, \ldots, \mathbf{Z}_{iT})$ determined by f_L^* for query $\mathbf{Q}_i$ is the same as that determined by $P\big(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it}\big)$ in that it holds with probability 1 that

$$\big(f_L^*(\mathbf{Q}_i, \mathbf{Z}_{it}) - f_L^*(\mathbf{Q}_i, \mathbf{Z}_{it'})\big) \big(P(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it}) - P(Y_{it'} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it'})\big) \ge 0, \quad (5)$$

for any $t \ne t' \in \{1, \ldots, T\}$, and the equality holds iff $P(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it}) = P(Y_{it'} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it'})$.

For Lemma 1, we denote f_L^* as f^* for any L defined in (1)–(4), and f^* is referred to as the Bayes ranker subsequently. One important metric to measure the ranking performance of a ranking function f is through its ranking regret relative to the Bayes ranker, defined as $\text{Reg}_L(f) = E(L(f, \boldsymbol{R})) - E(L(f^*, \boldsymbol{R})) \ge 0$.

It can be verified that $\text{Reg}_{L_{\text{sum}}}(f) = \text{Reg}_{L_{\text{pair}}}(f)$ as in Ailon (2014). In addition, the following Lemma 2 indicates that the ranking regrets defined by DCE and MAE can be upper-bounded by the pairwise regret.

Lemma 2 (Regret relations). For any ranking function f, if it has distinct values at $\mathbf{Z}_{it}$'s with probability 1, then

$$\text{Reg}_{L_{\text{mae}}}(f) \le \frac{T}{2} \text{Reg}_{L_{\text{pair}}}(f),$$
$$\text{Reg}_{L_{\text{dce}}}(f) \le T\Big(1 - \frac{1}{\log_2 3}\Big) \text{Reg}_{L_{\text{pair}}}(f).$$

A direct implication of Lemma 2 is that if a ranking function achieves small pairwise regret, the regrets defined by the other three losses are also small. Therefore, we construct a surrogate loss of $L_{\text{pair}}(f, \boldsymbol{R})$ as an operating loss for the proposed ranker. This is achieved by replacing the indicator function $I(f(\mathbf{Q}_i, \mathbf{Z}_{it}) \le f(\mathbf{Q}_i, \mathbf{Z}_{it'}))$ in $L_{\text{pair}}(f, \boldsymbol{R})$ by $\psi(f(\mathbf{Q}_i, \mathbf{Z}_{it}) - f(\mathbf{Q}_i, \mathbf{Z}_{it'}))$, where $\psi(s) = \min(1, (1 - s/\theta)_+)$ (Shen et al. 2003) is a scaled ψ-loss with a prespecified constant $\theta > 0$; see Figure 1 for a display of the scaled ψ-loss, the hinge loss and the zero-one loss.

The advantage of using $\psi(\cdot)$ is to increase the separation between the outcomes of 1 and 0 in ranking, while still approximating the indicator function. Specifically, the proposed surrogate loss function of (2) is

$$L_\psi(f, \boldsymbol{R}) = \frac{1}{NT^2} \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{t'=1}^{T} I(Y_{it} > Y_{it'}) \psi\big(f(\mathbf{Q}_i, \mathbf{Z}_{it}) - f(\mathbf{Q}_i, \mathbf{Z}_{it'})\big). \quad (6)$$

The following Lemma 3 shows that $L_\psi(\cdot, \cdot)$ retains the property of $L_{\text{pair}}(\cdot, \cdot)$.

Lemma 3 (The scaled ψ-loss). For any $\theta > 0$, $f_\psi^* = \text{argmin}_f E\big(L_\psi(f, \boldsymbol{R})\big)$ gives the same rank of $(\mathbf{Z}_{i1}, \ldots, \mathbf{Z}_{iT})$ for $\mathbf{Q}_i$ as that given by $P\big(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it}\big)$ in that it holds with probability 1 that

$$\big(f_\psi^*(\mathbf{Q}_i, \mathbf{Z}_{it}) - f_\psi^*(\mathbf{Q}_i, \mathbf{Z}_{it'})\big) \big(P(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it}) - P(Y_{it'} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it'})\big) \ge 0, \quad (7)$$

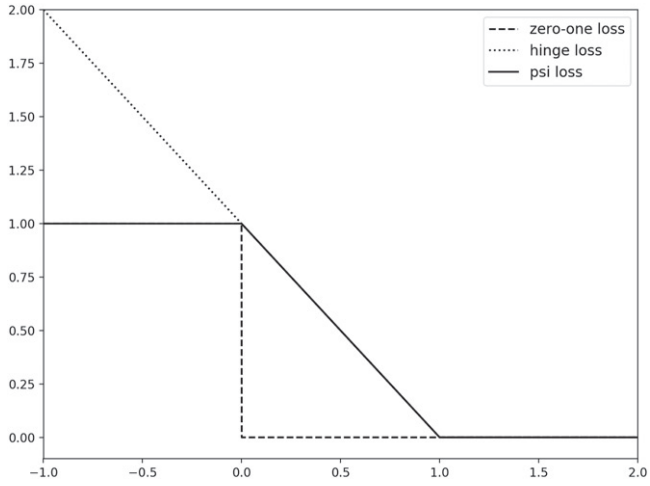

Figure 1. The zero-one loss, the hinge loss, and the scaled-ψ loss functions.

for any $t \neq t' \in \{1, \ldots, T\}$, and the equality holds iff $P(Y_{it} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it}) = P(Y_{it'} = 1 \mid \mathbf{Q}_i, \mathbf{Z}_{it'})$. Moreover, for any ranking function f,

$$\mathrm{Reg}_{L_{\mathrm{pair}}}(f) \leq \mathrm{Reg}_{L_\psi}(f), \tag{8}$$

and

$$\mathrm{Reg}_{L_{\mathrm{mae}}}(f) \leq \frac{T}{2}\mathrm{Reg}_{L_\psi}(f),$$
$$\mathrm{Reg}_{L_{\mathrm{dce}}}(f) \leq T\Big(1 - \frac{1}{\log_2 3}\Big)\mathrm{Reg}_{L_\psi}(f).$$

It is important to note that (8) does not hold for L_{svm} in general when the scaled ψ-loss in (6) is replaced by the hinge loss $(1 - s/\theta)_+$ although L_{svm} satisfies (7) (Uematsu and Lee 2017). This property is very different from classification. In fact, $\mathrm{Reg}_{L_{\mathrm{pair}}}(f) \leq \mathrm{Reg}_L^{1/2}(f)$ for exponential loss and logistic loss L (Agarwal 2014; Gao and Zhou 2015), which ultimately impedes the ranking performance if it is used as opposed to L_ψ.

In summary, $L_\psi(\cdot, \cdot)$ can be an appropriate surrogate loss for the losses in (1)–(4), with regrets upper-bounding the corresponding four regrets $\mathrm{Reg}_L(f)$, as showed in Lemmas 2 and 3. Consequently, a ranker f that performs well under L_ψ is expected to perform well under any other loss function L, which makes L_ψ desirable. Another benefit of L_ψ is that it is easier to implement while retaining desirable statistical properties as suggested in Theorem 1. On the other hand, the difference between L_{mae}, L_{dce}, and L_ψ lies in the weights $|\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}|^{-1}$. As showed in Lemma 3, the weights $|\mathcal{I}_{f(\mathbf{Q}_i, \mathbf{Z}_{it})}|^{-1}$ play a role of breaking ties in view of (5) and (7). Therefore, there is no efficiency loss when the weights are discarded in L_ψ.

2.2. Collaborative Ranking Function

To incorporate the collaborative information across queries and items, we propose a novel ranking function based on a latent factor model with a nested structure of user-specific queries. Specifically, for query $\boldsymbol{q}_i = (\boldsymbol{q}_{i1}, \ldots, \boldsymbol{q}_{iG})'$ with $\boldsymbol{q}_{ig} \in \{1, \ldots, n_g\}$ involving multiple queries associated with user i, we introduce one additional layer of latent factors $\mathbf{a}^g_{\boldsymbol{q}_{ig}}$ as a K-dimensional latent factor corresponding to $\boldsymbol{q}_{ig}$. Then the proposed ranking function becomes

$$f(\boldsymbol{q}_i, \mathbf{z}_{it}) = \boldsymbol{\beta}'\mathbf{z}_{it} + \sum_{g=1}^{G} \mathbf{b}_t'\mathbf{a}^g_{\boldsymbol{q}_{ig}}, \tag{9}$$

where G is the number of latent factors, $\boldsymbol{\beta}$ is an overall query-item effect coefficient of $\mathbf{z}_{it}$, and $\mathbf{a}_{g,\boldsymbol{q}_{ig}}$ and $\mathbf{b}_t$ are K-dimensional query-specific and item-specific latent factors, representing the intrinsic degree of preference of a user making query $\boldsymbol{q}_i$ and the intrinsic degree of relevance on item t.

The ranking function in (9) is highly interpretable. First, $f(\boldsymbol{q}_i, \mathbf{z}_{it})$'s tend to be positively correlated through the common latent factors. This reflects the fact that the responses Y_{it}'s tend to be positively associated when the corresponding queries share the same factors. Second, $f(\boldsymbol{q}_i, \mathbf{z}_{it})$ imposes a nested structure as a result of the components $\mathbf{a}^g_{\boldsymbol{q}_{ig}}$'s in (9). Third, as in mixed-effect models, $\boldsymbol{\beta}'\mathbf{z}_{it}$ can be regarded as the item-specific main effect, and $\mathbf{b}_t'\mathbf{a}^g_{\boldsymbol{q}_{ig}}$ as random effects representing interactions between users and items.

2.3. Proposed Method

In this subsection, we propose the collaborative ranking formulation based on a training set $(\boldsymbol{q}_i, \mathbf{z}_{it}, y_{it})_{i=1,\ldots,N;t=1,\ldots,T}$:

$$\tilde{L}_\psi(f, \boldsymbol{R}) = \frac{1}{NT^2}\sum_{i=1}^{N}\sum_{t',t} I(y_{it} > y_{it'})$$
$$\psi\big(f(\boldsymbol{q}_i, \mathbf{z}_{it}) - f(\boldsymbol{q}_i, \mathbf{z}_{it'})\big) + \lambda J(f), \tag{10}$$

where $J(f)$ is a nonnegative regularizer such as the inverse of the separation margin, Cortes and Vapnik (1995) and $\lambda \geq 0$ is a tuning parameter controlling the trade-off between the ranking of training data and the separation margin. Normally, let $\mathbf{a} = \big((\mathbf{a}^1)', \ldots, (\mathbf{a}^G)'\big)'$ with $\mathbf{a}^g = \big((\mathbf{a}^g_1)', \ldots, (\mathbf{a}^g_{n_g})'\big)'$, then $J(f) = \|\boldsymbol{\beta}\|^2 + \sum_{g=1}^{G}\|\mathbf{a}^g\|^2 + \sum_{t=1}^{T}\|\mathbf{b}_t\|^2$ is the squared L_2-norm, which is analogous to the notion of geometric separation in classification (Cortes and Vapnik 1995).

3. Scalable Computation

In this section, we develop a scalable computing scheme to solve (10). Specifically, the computing scheme employs a difference convex algorithm to relax nonconvex minimization (10) into a sequence of convex subproblems, where each subproblem is solved by the block successive upper-bound minimization (BSUM) algorithm (Razaviyayn, Hong, and Luo 2013) and a "divide-and-conquer" strategy. In this fashion, the scheme decomposes (10) into multiple small-scale tasks for parallelization and alleviates the memory requirement to achieve scalability.

To proceed, we introduce three blocks, namely, the overall effect $\boldsymbol{\beta}$, the item-specific latent factors denoted by $\mathbf{b}_1, \ldots, \mathbf{b}_T$, and the user-specific latent factors defined by $\mathbf{a}^1, \ldots, \mathbf{a}^G$. First, the scaled ψ-loss is decomposed by $\psi(u) = (1 - u/\theta)_+ - (-u/\theta)_+$, and thus (10) is rewritten as

$$\sum_{i=1}^{N}\sum_{t',t} y_{itt'}\big(1 - \vartheta_{itt'}/\theta\big)_+ - \sum_{i=1}^{N}\sum_{t',t} y_{itt'}\big(-\vartheta_{itt'}/\theta\big)_+$$
$$+\lambda\Big(\|\boldsymbol{\beta}\|^2 + \sum_{g=1}^{G}\|\mathbf{a}^g\|^2 + \sum_{t=1}^{T}\|\mathbf{b}_t\|^2\Big),$$

where $y_{itt'} = I(y_{it} > y_{it'})$, $\mathbf{z}_{itt'} = \mathbf{z}_{it} - \mathbf{z}_{it'}$, $\bar{\mathbf{a}}_i = \mathbf{a}^1_{\boldsymbol{q}_{i1}} + \cdots + \mathbf{a}^G_{\boldsymbol{q}_{iG}}$, and $\vartheta_{itt'} = \boldsymbol{\beta}'\mathbf{z}_{itt'} + \bar{\mathbf{a}}_i'\big(\mathbf{b}_t - \mathbf{b}_{t'}\big)$. Second, as in Liu, Shen, and Wong (2005), we linearize $\sum_{i=1}^{N}\sum_{t',t} y_{itt'}\big(-\vartheta_{itt'}/\theta\big)_+$ at a current solution $(\mathbf{a}_o, \mathbf{b}_o, \boldsymbol{\beta}_o)$, which leads to a convex upper bound of (10) $\tilde{L}^U_\psi\big(\mathbf{a}, \mathbf{b}, \boldsymbol{\beta}; \mathbf{a}_o, \mathbf{b}_o, \boldsymbol{\beta}_o\big)$ as follows:

$$\sum_{i=1}^{N}\sum_{t,t'} y_{itt'}\,(1 - \vartheta_{itt'}/\theta)_+ + \lambda\Big(\|\boldsymbol{\beta}\|^2 + \sum_{g=1}^{G}\|\mathbf{a}^g\|^2 + \sum_{t=1}^{T}\|\mathbf{b}_t\|^2\Big)$$
$$-\sum_{i=1}^{N}\sum_{t,t'} y_{itt'}\,\big(-\bar{\vartheta}_{itt'}/\theta\big)_+$$

$$+\frac{1}{\theta}\sum_{i=1}^{N}\sum_{t,t'} I(\bar{\vartheta}_{itt'}<0)\mathbf{z}'_{itt'}(\boldsymbol{\beta}-\boldsymbol{\beta}_o)$$
$$+\frac{1}{\theta}\sum_{i=1}^{N}\sum_{t,t'} I(\bar{\vartheta}_{itt'}<0)\bar{\mathbf{a}}'_{oi}\left(\mathbf{1}_{tt'}\otimes \boldsymbol{I}_{K\times K}\right)(\mathbf{b}-\mathbf{b}_o)$$
$$+\frac{1}{\theta}\sum_{g=1}^{G}\sum_{j=1}^{n_g}\sum_{i\in\mathcal{Q}_j^{(g)}}\sum_{t,t'} I(\bar{\vartheta}_{itt'}<0)(\mathbf{b}_{ot}-\mathbf{b}_{ot'})'(\mathbf{a}_{gj}-\mathbf{a}_{ogj}), \tag{11}$$

where $\bar{\vartheta}_{itt'}=\boldsymbol{\beta}'_o\mathbf{z}_{itt'}+\bar{\mathbf{a}}'_{oi}(\mathbf{b}_{ot}-\mathbf{b}_{ot'})$, $\mathbf{b}=(\mathbf{b}'_1,\ldots,\mathbf{b}'_T)'$, $\mathbf{1}_{tt'}$ is a T-dimensional vector with 1 at the tth element, -1 at the t'th element -1 and 0 otherwise, $\boldsymbol{I}_{K\times K}$ is a $K\times K$ matrix of all 1's, $\otimes$ stands for the Kronecker product, and $\mathcal{Q}_j^{(g)}=\{i:\boldsymbol{q}_{ig}=j\}$.

The BSUM algorithm minimizes (11) using blockwise descent iteratively, which solves the following three subproblems (12), (14), and (15) alternatively. Note that $\tilde{L}_\psi^U$ is blockwise convex, which ensures the unique minimizer for each of the subproblems.

3.1. Item-Effect Block

After fixing $(\mathbf{a}^1,\ldots,\mathbf{a}^G)'$ and $\boldsymbol{\beta}$, $(\mathbf{b}_1,\ldots,\mathbf{b}_T)'$ is updated by minimizing

$$\min_{\mathbf{b}}\sum_{i=1}^{N}\sum_{t,t'}\left(\xi_{itt'}+\frac{1}{\theta}I(\bar{\vartheta}_{itt'}<0)\bar{\mathbf{a}}'_i\left(\mathbf{1}_{tt'}\otimes\boldsymbol{I}_{K\times K}\right)(\mathbf{b}-\mathbf{b}_o)\right)+\lambda\mathbf{b}'\mathbf{b} \tag{12}$$
$$\text{subject to}\quad \vartheta_{itt'}\geq\theta(1-\xi_{itt'})\text{ and }\xi_{itt'}\geq 0,$$

with respect to $\mathbf{b}$ and $\xi_{itt'}$'s, where $\xi_{itt'}$'s are some slack variables.

3.2. User Effect Block

For each $\mathbf{a}^g$ with $g=1,\ldots,G$, we fix $\boldsymbol{\beta}$, $(\mathbf{b}_1,\ldots,\mathbf{b}_T)$ and $\mathbf{a}^{g'}$ with $g'\neq g$, and $\mathbf{a}^g$ can be updated by minimizing

$$\sum_{j=1}^{n_g}\sum_{i\in\mathcal{Q}_j^{(g)}}\sum_{t,t'}\Big(y_{itt'}(1-\vartheta_{itt'}/\theta)_+ \tag{13}$$
$$+\frac{1}{\theta}I(\bar{\vartheta}_{itt'}<0)(\mathbf{b}_t-\mathbf{b}_{t'})'(\mathbf{a}_j^g-\mathbf{a}_{o,j}^g)\Big)+\lambda\sum_{j=1}^{n_g}\|\mathbf{a}_j^g\|_2^2,$$

with respect to $\mathbf{a}^g$. Note that (13) can be solved separately with respect to $(\mathbf{a}_j^g)_{j=1}^{n_g}$. That is, for each $\mathbf{a}_j^g(j=1,\ldots,n_g)$, after introducing $\xi_{itt'}$, we solve (13) in a parallel fashion:

$$\min_{\mathbf{a}_{gj},\xi_{itt'}}\sum_{i\in\mathcal{Q}_j^{(g)}}\sum_{t,t'}\left(\xi_{itt'}+\frac{1}{\theta}I(\bar{\vartheta}_{itt'}<0)(\mathbf{b}_t-\mathbf{b}_{t'})'(\mathbf{a}_j^g-\mathbf{a}_{o,j}^g)\right)+\lambda\|\mathbf{a}_j^g\|_2^2,$$
$$\text{subject to}\quad \boldsymbol{\beta}'\mathbf{z}_{itt'}+(\mathbf{a}_j^g)'(\mathbf{b}_t-\mathbf{b}_{t'})+(\bar{\mathbf{a}}_i^{-g})'(\mathbf{b}_t-\mathbf{b}_{t'})$$
$$\geq\theta(1-\xi_{itt'})\quad\text{and}\quad\xi_{itt'}\geq 0, \tag{14}$$

where $\bar{\mathbf{a}}_i^{-g}=\sum_{g'\neq g}\mathbf{a}_{\boldsymbol{q}_{ig'}}^{g'}$. This computational strategy is summarized in Figure 2.

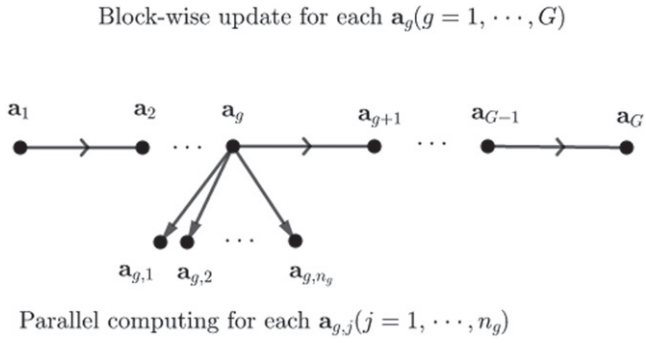

Figure 2. The chart on the scheme of updating user effect.

3.3. Main Effect Block

After fixing $(\mathbf{a}^1,\ldots,\mathbf{a}^G)$ and $(\mathbf{b}_1,\ldots,\mathbf{b}_T)$, we update $\boldsymbol{\beta}$ by solving

$$\min_{\boldsymbol{\beta}}\sum_{i=1}^{N}\sum_{t,t'}\left(\xi_{itt'}+\frac{1}{\theta}I(\bar{\vartheta}_{itt'}<0)\mathbf{z}'_{itt'}(\boldsymbol{\beta}-\boldsymbol{\beta}_o)\right)+\lambda\|\boldsymbol{\beta}\|_2^2, \tag{15}$$
$$\text{subject to }\boldsymbol{\beta}'\mathbf{z}_{itt'}+\bar{\mathbf{a}}'_i(\mathbf{b}_t-\mathbf{b}_{t'})\geq\theta(1-\xi_{itt'})\text{ and }\xi_{itt'}\geq 0.$$

Note that only the user effect can be parallelizable in the BSUM algorithm. The above process can be summarized as in Algorithm 1, where the final solution is denoted as $(\widehat{\mathbf{a}}^1,\ldots,\widehat{\mathbf{a}}^G)$, $(\widehat{\mathbf{b}}_1,\ldots,\widehat{\mathbf{b}}_T)$, and $\widehat{\boldsymbol{\beta}}$, and thus $\widehat{f}(\boldsymbol{q}_i,\mathbf{z}_{it})=\widehat{\boldsymbol{\beta}}'\mathbf{z}_{it}+\sum_{g=1}^{G}\widehat{\mathbf{b}}'_t\widehat{\mathbf{a}}_{\boldsymbol{q}_{ig}}^g$. Finally, the items $\mathbf{z}_{i1},\ldots,\mathbf{z}_{iT}$ for each specific query $\boldsymbol{q}_i$ can be ranked based on the values of $\widehat{f}(\boldsymbol{q}_i,\mathbf{z}_{it})$.

Algorithm 1 (Parallel BSUM algorithm). Step 1 (Initialization): Initialize $(\mathbf{a}_o^g)_{g=1}^G$, $\boldsymbol{\beta}_o$ and $\mathbf{b}_o$. Specify a tolerance level.

Step 2 (Item-effect): Update $\mathbf{b}_o$ by solving (12) with respect to $\mathbf{b}$ with $\mathbf{a}$ and $\boldsymbol{\beta}$ fixed.

Step 3 (Parallelization for user-effect): Update $\mathbf{a}_o^g$ in a parallel fashion by solving (14) for each $\mathbf{a}_j^g$ with fixed $\mathbf{b}$, $\boldsymbol{\beta}$ and $\mathbf{a}_{g'}$ with $g'\neq g$.

Step 4 (Main-effect): Update $\boldsymbol{\beta}_o$ by solving (15) for $\boldsymbol{\beta}$ with fixed $\mathbf{a}$ and $\mathbf{b}$.

Step 5 (Termination): Iterate Steps 2–4 until the decrement of the objective function is less than the tolerance level.

The following lemma shows that the BSUM algorithm converges to a coordinate-wise minimizer Razaviyayn, Hong, and Luo (2013) of (10), which cannot be further improved by any coordinate-wise updates.

Lemma 4 (Convergence). Any solution $(\widehat{\mathbf{a}},\widehat{\mathbf{b}},\widehat{\boldsymbol{\beta}})$ obtained from Algorithm 1 is the coordinate-wise minimizer of (10).

Convergence of the BSUM algorithm is ensured by Lemma 4. In fact, if we further assume (10) is regular at $(\widehat{\mathbf{a}},\widehat{\mathbf{b}},\widehat{\boldsymbol{\beta}})$, then it follows from Razaviyayn, Hong, and Luo (2013) that $(\widehat{\mathbf{a}},\widehat{\mathbf{b}},\widehat{\boldsymbol{\beta}})$ must be a local minimizer of (10) as well.

4. Theory

This section explains the theoretical properties of the proposed collaborative ranking method in terms of its ranking accuracy,

measure by the pairwise regret. Particularly, an error bound for the pairwise regret is established with regard to the number of queries and items, the factors of query, and tuning parameters. Specifically, the function space $\mathcal{F}$ of candidate ranking functions in (9) can be written as

$$\mathcal{F} = \Big\{ f(\boldsymbol{q}_i, \mathbf{z}_{it}) = \boldsymbol{\beta}'\mathbf{z}_{it} + \sum_{g=1}^{G} \mathbf{b}_t' \mathbf{a}^g_{q_{ig}}, \quad \text{for } i = 1, \ldots, N;$$
$$t = 1, \ldots, T : \boldsymbol{\beta} \in \mathbb{R}^p; \mathbf{a}^g_j \in \mathbb{R}^K, \quad \text{for } g = 1, \ldots, G;$$
$$j = 1, \ldots, n_g; \mathbf{b}_t \in \mathbb{R}^K, \quad \text{for } t = 1, \ldots, T \Big\},$$

and $\gamma = p + (T + \sum_{g=1}^{G} n_g)K$ denotes the total number of parameters used in $\mathcal{F}$. The following technical assumptions are made:

Assumption A (Bounded moment). Assume that $N^{-1}\sum_{i=1}^{N} E(\|\mathbf{Z}_{i1}\|_2^4)$ is upper bounded by $c_{\mathbf{Z}}$ for all N.

Assumption B (Variance property). There exist constants $\kappa > 0$, $c_1 > 0$, and small $\epsilon_0 > 0$, such that for any $0 \le \epsilon \le \epsilon_0$ and $f \in \mathcal{F}$,

$$\sup_{f \in \mathcal{F}: \mathrm{Reg}_{L_\psi}(f) \le \epsilon} \frac{1}{N} \sum_{i=1}^{N} \mathrm{var}\left(L_\psi(f^\star, \boldsymbol{R}_{\mathbf{Q}_i}) - L_\psi(f, \boldsymbol{R}_{\mathbf{Q}_i}) \right) \le c_1 \epsilon^\kappa.$$

Assumption A imposes a moment condition on predictors $\mathbf{Z}_{it}$, and Assumption B is a local smoothness condition on the variance of the criterion function difference in the neighborhood of $f^\star$. For example, Assumption B is satisfied with $\kappa = 1$ if $\mathbf{Z}_{i1}$ follows a discrete distribution and there exists a constant c_2 such that $\left|\eta(\mathbf{z}, \boldsymbol{q}) - \eta(\mathbf{z}', \boldsymbol{q})\right| \ge c_2$ for any $\boldsymbol{q}$, where $\eta(\mathbf{z}, \boldsymbol{q}) = P(Y_{i1} = 1 | \mathbf{Z}_{i1} = \mathbf{z}, \mathbf{Q}_i = \boldsymbol{q})$.

Theorem 1. If Assumptions A and B hold, and $N \ge c_3\gamma$ for some constant $c_3 > 0$, then for any minimizer $\widehat{f}$ of (10), there exists a constant $c_4 > 0$ such that

$$P\big(\mathrm{Reg}_{L_{\mathrm{pair}}}(\widehat{f}) \ge \epsilon_N^2\big) \le 3\exp\big(- c_4 N (\lambda_N J_0)^{2-\min(1,\kappa)}\big) + 2\exp\big(-c_4 N \epsilon_0^2\big), \tag{16}$$

given that $\epsilon_N^2 = O\big(\frac{\gamma}{N}\log(\frac{N}{\gamma})\big)^{\frac{1}{2-\min(1,\kappa)}} \le \epsilon_0$, when $\lambda_N \sim \epsilon_N^2$. Here $J_0 = \max(1, J(f^\star))$, $\gamma = p + (T + \sum_{g=1}^{G} n_g)K$ is the total number of parameters, and c_3 is a constant depending on δ_0, $c_{\mathbf{Z}}$, and J_0. As a result, $\mathrm{Reg}_{L_{\mathrm{pair}}}(\widehat{f}) = O_p\big(\frac{\gamma}{N}\log(\frac{N}{\gamma})\big)^{\frac{1}{2-\min(1,\kappa)}}$.

Theorem 1 suggests that the convergence rate of the pairwise regret of $\widehat{f}$ is of order $\big(\frac{\gamma}{N}\log(\frac{N}{\gamma})\big)^{\frac{1}{2-\min(1,\kappa)}}$. When $\kappa = 1$, this rate becomes $\frac{\gamma}{N}\log(\frac{N}{\gamma})$, matching up with the fast classification rate (Shen et al. 2003; Bartlett, Jordan, and McAuliffe 2006). Furthermore, together with Lemma 2, it implies the following corollary, which provides a rate of convergence for other bipartite ranking losses.

Corollary 1. Under the assumptions of Theorem 1, (16) holds uniformly over four other losses, or L can be $L_{\mathrm{pair}}, L_{\mathrm{mae}}, L_{\mathrm{dce}}$, or L_{sum}. Moreover, $\mathrm{Reg}_L(\widehat{f}) = O_p\Big(\frac{T\gamma}{N}\log(\frac{N}{\gamma})\Big)^{\frac{1}{2-\min(1,\kappa)}}$ when L is L_{mae} or L_{dce}; and $\mathrm{Reg}_L(\widehat{f}) = O_p\Big(\frac{\gamma}{N}\log(\frac{N}{\gamma})\Big)^{\frac{1}{2-\min(1,\kappa)}}$ when L is L_{pair} or L_{sum}.

5. Numerical Examples

This section examines the operating characteristics of the proposed ψ-ranker with $\theta = 1$ in simulated and real examples, and compares it with strong competitors in the literature, including the support vector machine (SVM; Cortes and Vapnik 1995), pairwise ranking with SVM (pairwise-SVM; Joachims 2002), regularized single factor singular value decomposition for recommender systems (rSVD; Paterek 2007), and a ReLU neural network logistic regression (neural network; Glorot, Bordes, and Bengio 2011). Specifically, for SVM and neural network, we convert a categorical covariate to a binary vector with the one-hot encoding scheme, and rank items by values of the estimated conditional class probabilities. The conditional class probabilities are estimated by the values of the last layer of the neural network.

For implementation, we code Algorithm 1 using the Python package VarSVM,[1] SVM based on the package linearSVC,[2] rSVD based on Dai et al. (2019), and neural networks based on package MLPClassifier.[3] To facilitate the evaluation, we report the MAE for the top 3 ranked items for each query, as suggested in Herlocker et al. (2004).

5.1. Simulated Examples

One simulated example $\{\boldsymbol{q}_i, \mathbf{z}_{it}, y_{it}\}_{i=1,\ldots,N;t=1,\ldots,T}$ is generated to mimic the situation of hotel booking. Specifically, $\boldsymbol{q}_i$ is the ith discrete-valued query vector, $\mathbf{z}_{it} \in \mathbb{R}^p$ is a vector of item predictors, and $y_{it} \in \{0, 1\}$ with 1 indicating the preference of item t for query $\boldsymbol{q}_i$ and 0 otherwise.

First, given a vector $(n_1, \ldots, n_G)$, a total of N queries $\big\{\boldsymbol{q}_i = (\boldsymbol{q}_{i1}, \ldots, \boldsymbol{q}_{iG})'\big\}_{i=1}^{N}$ are generated, where $\boldsymbol{q}_{ig}$ is uniformly sampled from $\{1, \ldots, n_g\}$ to mimic the discrete-valued query of hotel booking, such as destination id, the number of rooms, and the length of stay, and G denotes the number of query options. Next, $\boldsymbol{z}_{it} = \boldsymbol{\omega}_t$; $i = 1, \ldots, N$; $t = 1, \ldots, T$, which may be thought of as a hotel feature that remains unchanged for each query and independent of t, where $\boldsymbol{\omega}_t$ is independently generated from $N(0, \sqrt{0.1}\boldsymbol{I}_d)$ for the item covariate, such as hotel price and hotel review, and $\boldsymbol{\beta}$ is generated from $N(0, \boldsymbol{I}_d)$, $\mathbf{a}^g_j$ and $\mathbf{b}_t$ are generated from $N(0, \boldsymbol{I}_{K^0})$ to simulate the heterogeneity among users and items, where K^0 is the number of latent factors. For each query $\boldsymbol{q}_i$, we generate a preference score $s_{it} = \boldsymbol{\beta}'\mathbf{z}_{it} + (\mathbf{a}^1_{q_{i1}} + \cdots + \mathbf{a}^G_{q_{iG}})'\mathbf{b}_t + \varepsilon_t$ for each item, where $\varepsilon_t \sim N(0, \sqrt{0.1})$. As in the hotel booking example, where each query results in only one booked hotel, we set the item with the highest score as the "booked hotel" with $Y_{it} = 1$, and set $Y_{it'} = 0$ for all other items. Finally, the generated dataset is split into training, validation and testing sets with proportions of 50%, 20% and 30%, respectively. Then a ranker is fitted on the training set, is tuned on the validation set, and is evaluated by MAE (4) on the test set.

[1] https://github.com/statmlben/Variant-SVM
[2] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
[3] https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

Table 1. Averaged MAEs and running time in seconds of various methods and their standard deviations in parenthesis over 100 replications.

	Bayes	ψ-Ranker	SVM	Pairwise-SVM	rSVD	Neural network
$N = 200$						
$K^0 = 5$	0.098(0.008)	0.284(0.020)	0.430(0.032)	0.388(0.029)	0.371(0.028)	0.274(0.019)
$K^0 = 10$	0.077(0.006)	0.314(0.017)	0.493(0.029)	0.453(0.026)	0.443(0.022)	0.309(0.018)
$K^0 = 20$	0.057(0.005)	0.342(0.015)	0.582(0.023)	0.543(0.022)	0.518(0.020)	0.342(0.015)
Time (sec)	–	0.985(0.033)	0.036(0.001)	0.003(0.000)	0.322(0.047)	0.636(0.018)
$N = 500$						
$K^0 = 5$	0.099(0.006)	0.194(0.011)	0.387(0.023)	0.341(0.023)	0.326(0.019)	0.214(0.013)
$K^0 = 10$	0.075(0.004)	0.212(0.009)	0.531(0.024)	0.473(0.022)	0.447(0.021)	0.240(0.014)
$K^0 = 20$	0.055(0.003)	0.218(0.010)	0.504(0.021)	0.477(0.020)	0.454(0.019)	0.259(0.010)
Time (sec)	–	2.37(0.078)	0.109(0.006)	0.004(0.000)	0.543(0.065)	1.11(0.086)
$N = 1000$						
$K^0 = 5$	0.090(0.006)	0.152(0.009)	0.423(0.028)	0.372(0.023)	0.364(0.023)	0.180(0.012)
$K^0 = 10$	0.100(0.007)	0.174(0.010)	0.449(0.027)	0.391(0.026)	0.370(0.023)	0.199(0.013)
$K^0 = 20$	0.057(0.002)	0.157(0.006)	0.540(0.021)	0.496(0.020)	0.470(0.020)	0.221(0.010)
Time (sec)	–	4.09(0.110)	0.267(0.005)	0.006(0.000)	0.901(0.023)	1.61(0.048)
$N = 2000$						
$K^0 = 5$	0.103(0.006)	0.148(0.009)	0.497(0.031)	0.415(0.025)	0.392(0.023)	0.191(0.012)
$K^0 = 10$	0.075(0.003)	0.126(0.006)	0.507(0.023)	0.460(0.023)	0.428(0.020)	0.201(0.010)
$K^0 = 20$	0.053(0.002)	0.106(0.004)	0.550(0.027)	0.504(0.022)	0.478(0.021)	0.187(0.007)
Time (sec)	–	8.15(0.160)	0.615(0.008)	0.010(0.000)	1.13(0.008)	2.60(0.089)
$N = 3000$						
$K^0 = 5$	0.093(0.006)	0.126(0.009)	0.445(0.033)	0.362(0.024)	0.336(0.022)	0.167(0.012)
$K^0 = 10$	0.074(0.003)	0.113(0.004)	0.502(0.020)	0.453(0.018)	0.425(0.017)	0.189(0.007)
$K^0 = 20$	0.054(0.002)	0.093(0.003)	0.515(0.022)	0.480(0.020)	0.453(0.018)	0.181(0.006)
Time (sec)	–	12.7(0.564)	0.123(0.037)	0.014(0.001)	1.27(0.077)	3.92(0.179)

NOTE: Here Bayes, ψ-rank, SVM, pairwise-SVM, rSVD, and neural network denote the Bayes ranker, the proposed ψ-ranker, linear SVM classification, pairwise ranking with linear SVM, the regularized singular value decomposition, and the neural network.

In simulations, we fix $d = 10$, $T = 10$, $G = 3$, $n_1 = 50$, $n_2 = 10$, $n_3 = 2$, and consider $N = 200, 500, 1000, 2000, 3000$, and $K^0 = 5, 10, 20$.

For the rSVD, it uses continuous covariates $\mathbf{z}_{it}$, the first column of $\boldsymbol{q}$ and the item id t as latent factors. For the pairwise-SVM, only $\mathbf{z}_{it}$ is included, since $\boldsymbol{q}_i$ is eliminated for each item pair. For the SVM and neural network, all covariates are included, and $\boldsymbol{q}_i$ and item id t are numerically converted to multiple 0–1 dummy variables. We set $K = K^0$ for the proposed ψ-ranker and sSVD. For tuning λ, we perform a grid search over {0.01, 0.1, 1, 10, 20, 50, 100, 150, 200} on the validation set to seek the optimal λ to minimize the test error for the SVM, pairwise-SVM, rSVD, and ψ-ranker. For the neural network, we fix the number of nodes as 64 at each hidden layer, and tune the depth of hidden layers in a grid set {1, 2, 3, 4, 5, 6}. The MAEs and running times of each method on the testing set over 100 replications are summarized in Table 1, where the running times are provided for the SVM, pairwise-SVM, rSVD, and ψ-ranker with $\lambda = 0.01$, and for the neural network with 6 hidden layers.

As suggested in Table 1, the ψ-ranker outperforms the other competitors in terms of MAE, expect the proposed ψ-ranker and neural network share the similar performance when $n = 200$. The largest percentages of improvements are about 81.9%, 74.3%, 72.3%, and 48.6% over the SVM, pairwise-SVM, rSVD, and neural network, respectively. However, the proposed method is slightly slower than the other competitors. Furthermore, as suggested by Figure 3, the ψ-ranker has a faster rate of convergence than the neural network logistic regression, with the convergence power nearly twice, which is consistent with the theoretical result of Lemma 3 and Theorem 1. By comparison, the SVM, pairwise-SVM, and rSVD may not converge in terms of n.

5.2. Application to the Expedia Data Challenge

This section applies the ψ-ranker, pairwise-SVM, SVM, rSVD, and neural network to analyze a dataset for personalized predication of Expedia hotel searches in the Kaggle competition.[4] The Expedia data contain online user-specific records of shopping and purchasing, price competitiveness-related information, as well as users' responses on whether a hotel is booked. Specifically, each record includes variables: search id, date time, destination id, length of stay, children count, hotel review score, hotel location score, hotel rating, hotel price, hotel id, distance, is randomly sort, is book, where variable "distance" is the distance between the user's location and a hotel, and variable "is

[4] *https://www.kaggle.com/c/expedia-personalized-sort*

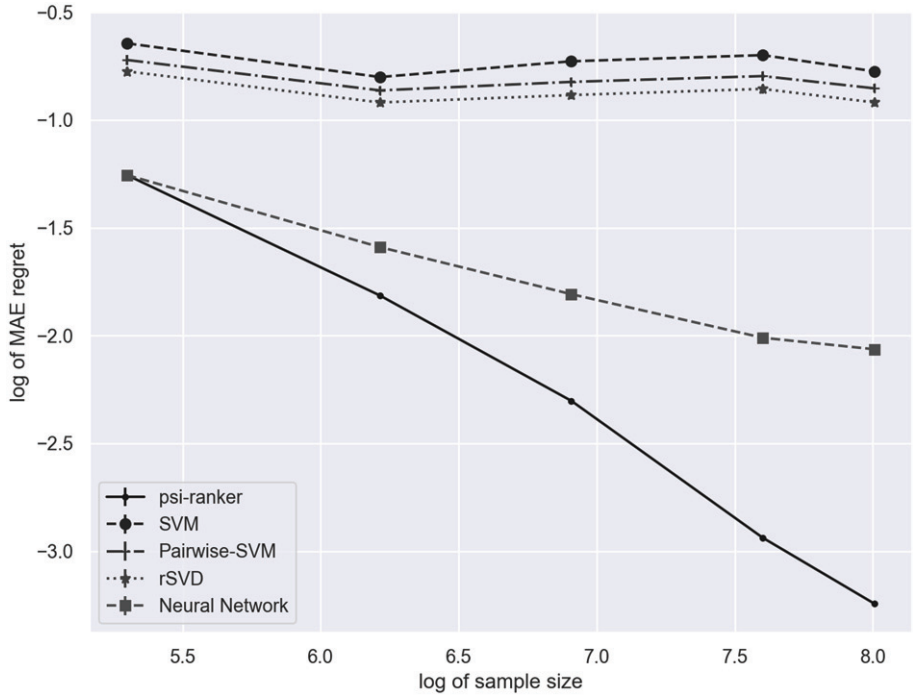

Figure 3. Logarithm of averaged MAE regrets of various methods in the simulated examples with $K^0 = 20$ as a function of logarithm of the sample size $n = 200, 500, 1000, 2000, 3000$, where the slope of the curve indicates the convergence power α for a method to yield a rate of n^{α}. Here ψ-ranker, SVM, pairwise-SVM, rSVD, and neural network denote the Bayes ranker, the ψ-ranker, linear SVM classification, pairwise ranking with linear SVM, the regularized singular value decomposition, and the neural network logistic regression.

Table 2. Averaged MAEs and running time in seconds of various methods and their standard deviations in parenthesis on 10% random subset of *Expedia* dataset over 50 replications.

	ψ-Ranker	SVM	Pairwise-SVM	rSVD	Neural network
MAE	0.377(0.001)	0.506(0.002)	0.396(0.002)	0.404(0.001)	0.406(0.001)
Time (sec)	612(57.8)	68.4(0.237)	35.7(0.100)	68.7(1.54)	2451(255)

NOTE: Here ψ-rank, SVM, rSVD, and neural network denote the Bayes ranker, the proposed ψ-ranker, linear SVM classification, the regularized singular value decomposition, and the neural network logistic regression.

randomly sort" is binary with 1/0 indicating random/normal displayed sort. Finally, "is book" indicates if the hotel is booked for a researching record.

Our goal is to predict if a hotel will be booked given each query. First, we extract query features "month" and "is Saturday" from date time. Second, for the proposed ψ-ranker, continuous covariates $\mathbf{z}_{it}$ include "hotel rating," "hotel review score," "hotel location score," "distance," "price," "length of stay," "children count," while the latent factor $\mathbf{a}$, in (9) and (10), consists of the factors of $\boldsymbol{q}_i$ as "month," "is Saturday," "is randomly sort" and the latent factors $\mathbf{b}$ there represent all "hotel id." Then, $y_{it} = 1$ for the booked hotels, and $y_{it} = -1$ for the view-only hotels under same "search id." For rSVD, it is infeasible to treat each query as one latent factor, so rSVD uses continuous covariates $\mathbf{z}_{it}$ and "month," "hotel id" as latent factors. For the pairwise-SVM, SVM, and neural network, all covariates are included, and each categorical covariate is numerically converted to multiple 0–1 dummy variables except for "hotel id" with levels over 60,000.

For evaluation, the MAEs and running times of each method over 50 replications are summarized in Table 2, where the running times are provided for SVM, rSVD, and ψ-ranker with $\lambda = 0.01$, and for neural network with 6 hidden layers. For each replication, we randomly sample 10% of the Expedia data in the United States, and split it into training, validation, and testing sets, with partition ratios of 50%, 20%, and 30%, respectively. The combined dataset consists of over 550,000 records from over 200,000 search queries involving 60,000 hotels. For the proposed ψ-ranker, the number of latent factors is chosen to be $K = 3$. For tuning λ, we perform a grid search over {0.01, 0.1, 1, 10, 20, 50, 100, 150, 200} on the validation set to seek the optimal λ minimizing the test error for SVM, pairwise-SVM, rSVD and ψ-ranker. For neural network, we fix the number of nodes in each hidden layer as 64, and tune the depth of hidden layers in a grid set {1, 2, 3, 4, 5, 6}.

As suggested in Table 2, the ψ-ranker again provides superior performance in terms of the MAEs. The percentages of improvements of the ψ-ranker over SVM, pairwise-SVM, rSVD, and neural network are 25.5%, 4.80%, 6.68%, and 7.14%, respectively. As expected, the pairwise ψ-loss contributes to the improved performance as well as the information of search queries and items incorporated by the nested structure of latent factors.

6. Discussion

This article develops the ψ-ranker, a new collaborative ranking system, which provides a user-specific list of most-preferred items by incorporating the information from users, items, and search queries collaboratively. Moreover, we prove that a sharper convergence of the ψ-ranker is achieved under four popular bipartite ranking losses simultaneously, including the sum loss, pairwise zero-one loss, DCE, and MAE. Computationally, two levels of nonconvex optimization are implemented via difference convex programming and block successive upper-bound minimization. Theoretically, we establish a probabilistic error bound for the ψ-ranker and show that the ranking error of the ψ-ranker has a sharper rate of convergence in the general framework of bipartite ranking losses, even when the number of the model parameters diverges with the sample size.

Supplementary Materials

The supplementary materials provide additional proofs of theorems and the codings used in simulations and real data application.

Acknowledgments

The authors thank the editors, the associate editor, and two anonymous referees for helpful comments and suggestions.

Funding

Research supported in part by NSF grants DMS-1712564, DMS-1721216, DMS-1613190, and Hong Kong RGC grants GRF-11331016, GRF-11303918, and GRF-11300919.

References

Agarwal, S. (2014), "Surrogate Regret Bounds for Bipartite Ranking via Strongly Proper Losses," *Journal of Machine Learning Research*, 15, 1653–1674. [2,4]

Ailon, N. (2014), "Improved Bounds for Online Learning Over the Permutahedron and Other Ranking Polytopes," in *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, pp. 29–37. [2,3]

Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. (2006), "Convexity, Classification, and Risk Bounds," *Journal of the American Statistical Association*, 101, 138–156. [6]

Billsus, D., and Pazzani, M. J. (2000), "User Modeling for Adaptive News Access," *User Modeling and User-Adapted Interaction*, 10, 147–180. [1]

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. (2005), "Learning to Rank Using Gradient Descent," in *Proceedings of the 22nd International Conference on Machine Learning*, pp. 89–96. [2]

Burke, R. (2002), "Hybrid Recommender Systems: Survey and Experiments," *User Modeling and User-Adapted Interaction*, 12, 331–370. [1]

Calauzenes, C., Usunier, N., and Gallinari, P. (2012), "On the (Non-)Existence of Convex, Calibrated Surrogate Losses for Ranking," in *Advances in Neural Information Processing Systems*, pp. 197–205. [2]

Chaudhuri, S., and Tewari, A. (2017), "Online Learning to Rank with Top-k Feedback," *Journal of Machine Learning Research*, 18, 1–50. [2]

Clémençon, S., Lugosi, G., and Vayatis, N. (2008), "Ranking and Empirical Minimization of U-Statistics," *The Annals of Statistics*, 36, 844–874. [2,3]

Cortes, C., and Vapnik, V. (1995), "Support-Vector Networks," *Machine Learning*, 20, 273–297. [4,6]

Dai, B., Wang, J., Shen, X., and Qu, A. (2019), "Smooth Neighborhood Recommender Systems," *Journal of Machine Learning Research*, 20, 589–612. [6]

Gao, W., and Zhou, Z. H. (2015), "On the Consistency of AUC Pairwise Optimization," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*. [4]

Glorot, X., Bordes, A., and Bengio, Y. (2011), "Deep Sparse Rectifier Neural Networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323. [6]

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004), "Evaluating Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems*, 22, 5–53. [6]

Joachims, T. (2002), "Optimizing Search Engines Using Click Through Data," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 133–142. [6]

Kotlowski, W., Dembczynski, K. J., and Huellermeier, E. (2011), "Bipartite Ranking Through Minimization of Scoring Loss," in *Proceedings of the 28th International Conference on Machine Learning*, pp. 1113–1120. [2]

Lang, K. (1995), "Newsweeder: Learning to Filter Netnews," in *Proceedings of the 12th International Conference on Machine Learning*, pp. 331–339. [1]

Li, Z., Uschmajew, A., and Zhang, S. (2015), "On Convergence of the Maximum Block Improvement Method," *SIAM Journal on Optimization*, 25, 210–233. []

Linden, G., Smith, B., and York, J. (2003), "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," *IEEE Internet Computing*, 7, 76–80. [1]

Liu, S., Shen, X., and Wong, W. H. (2005), "Computational Developments of ψ-Learning," in *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 1–11. [4]

Mao, X., Chen, S. X., and Wong, R. K. (2019), "Matrix Completion With Covariate Information," *Journal of the American Statistical Association*, 114, 198–210. [1]

Middleton, S. E., Shadbolt, N. R., and De Roure, D. C. (2004), "Ontological User Profiling in Recommender Systems," *ACM Transactions on Information Systems*, 22, 54–88. [1]

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., and Riedl, J. (2003), "MovieLens Unplugged: Experiences With an Occasionally Connected Recommender System," in *Proceedings of the 8th International Conference on Intelligent User Interfaces*, pp. 263–266. [1]

Paterek, A. (2007), "Improving Regularized Singular Value Decomposition for Collaborative Filtering," in *Proceedings of KDD Cup and Workshop*, pp. 5–8. [1,6]

Razaviyayn, M., Hong, M., and Luo, Z. Q. (2013), "A Unified Convergence Analysis of Block Successive Minimization Methods for Nonsmooth Optimization," *SIAM Journal on Optimization*, 23, 1126–1153. [4,5]

Salakhutdinov, R., Mnih, A., and Hinton, G. (2007), "Restricted Boltzmann Machines for Collaborative Filtering," in *Proceedings of the 24th International Conference on Machine Learning*, pp. 791–798. [1]

Shen, X., Tseng, G. C., Zhang, X., and Wong, W. H. (2003), "On ψ-Learning," *Journal of the American Statistical Association*, 98, 724–734. [3,6]

Uematsu, K., and Lee, Y. (2017), "On Theoretically Optimal Ranking Functions in Bipartite Ranking," *Journal of the American Statistical Association*, 112, 1311–1322. [4]

Valizadegan, H., Jin, R., Zhang, R., and Mao, J. (2009), "Learning to Rank by Optimizing NDCG Measure," in *Advances in Neural Information Processing Systems*, pp. 1883–1891. [2]

Wang, H., Wang, N., and Yeung, D. Y. (2015), "Collaborative Deep Learning for Recommender Systems," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 1235–1244. [1]

Yue, Y., Finley, T., Radlinski, F., and Joachims, T. (2007), "A Support Vector Method for Optimizing Average Precision," in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 271–278. [2]