

Singapore Management University

Institutional Knowledge at Singapore Management University

Research Collection School Of Computing and Information Systems

School of Computing and Information Systems

10-2019

Collaborative online ranking algorithms for multitask learning

Guangxia LI

Peilin ZHAO

Tao MEI

Peng YANG

Yulong SHEN

See next page for additional authors

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research



Part of the [Databases and Information Systems Commons](#), and the [Theory and Algorithms Commons](#)

Citation

LI, Guangxia; ZHAO, Peilin; MEI, Tao; YANG, Peng; SHEN, Yulong; CHANG, Julian K. Y.; and HOI, Steven C. H.. Collaborative online ranking algorithms for multitask learning. (2019). *Knowledge and Information Systems*. 62, (6), 2327-2348.


Available at: https://ink.library.smu.edu.sg/sis_research/5131

This Journal Article is brought to you for free and open access by the School of Computing and Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Computing and Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

Author

Guangxia LI, Peilin ZHAO, Tao MEI, Peng YANG, Yulong SHEN, Julian K. Y. CHANG, and Steven C. H. HOI

Collaborative online ranking algorithms for multitask learning

Guangxia Li¹  · Peilin Zhao² · Tao Mei³ · Peng Yang⁴ · Yulong Shen¹ · Julian Kuiyu Chang⁵ · Steven C. H. Hoi⁶

Abstract

There are many applications in which it is desirable to rank or order instances that belong to several different but related problems or tasks. Although unique, the individual ranking problem often shares characteristics with other problems in the group. Conventional ranking methods treat each task independently without considering the latent commonalities. In this paper, we study the problem of learning to rank instances that belong to multiple related tasks from the multitask learning perspective. We consider a case in which the information that is learned for a task can be used to enhance the learning of other tasks and propose a collaborative multitask ranking method that learns several ranking models for each of the related tasks together. The proposed algorithms operate in rounds by learning models from a sequence of data instances one at a time. In each round, our algorithms receive an instance that belongs to a task and make a prediction using the task's ranking model. The model is then updated by leveraging both the task-specific data and the information provided by other models in a collaborative way. The experimental results demonstrate that our algorithms can improve the overall performance of ranking multiple correlated tasks collaboratively. Furthermore, our algorithms can scale well to large amounts of data and are particularly suitable for real-world applications in which data arrive continuously.

Keywords Learning to rank · Online learning · Multitask learning

1 Introduction

Ranking or ordering objects according to their degrees of relevance, preference or importance has been extensively studied in data mining and information retrieval for decades. It has been applied to a wide range of real-world applications such as web search, document retrieval, online advertising, collaborative filtering and machine translation [20]. Despite being under the same guise of ranking, a ranking scheme can vary among applications. Consider web search ranking as an example: Given a query, it has to sort quite an amount of candidates (e.g., hundreds of thousands of pages) in a short time (e.g., hundreds of milliseconds), with special attention paid to making a few top-ranked results appealing to the user. By comparison, for rating products/services based on consumer feedback, the scale of ranks is typically much

smaller (e.g., rating movies as *poor*, *average*, *good*, *very good* and *excellent* or, equivalently, with one to five stars) and correct assignment of the rank labels is critical. A technique called ordinal regression (also known as ordinal classification) is particularly suitable for this task. In contrast to product ratings that deal with ordinal grades, a recommendation system ranks a collection of items (e.g., movies or products) based on the implicit feedback from users (e.g., click logs of a Web site or purchase records of customers) [25]. Collaborative filtering is dedicated to such problems.

An active line of research is the application of machine learning techniques to ranking problems. The *learning-to-rank* methods examine how a ranking model can be automatically constructed from training data, which consist of a set of input instances and corresponding ranks, such that the model can sort new instances in a reasonable way [24]. They are typically formulated as supervised learning processes. A learning algorithm is employed to learn the ranking model (i.e., the way of combining input features) such that the model can predict the ground-truth labels or orders in the training set as accurately as possible. The accuracy of prediction is typically measured by a loss function. A learning process is used to find an optimal solution to an optimization problem that involves the loss function. The trained model can be used to determine the rank labels or orders of unseen instances. Compared with heuristic approaches, learning-to-rank methods can combine many features and tune parameters automatically; hence, they are ideal for real-world applications with large amounts of data.

In this paper, we focus on a subproblem of learning-to-rank: We do not derive a relative order among items; instead, we assign a rank label from a set of predefined ordinal categories to each item. (An intuitive example is product rating, as discussed above.) When applied to multiple related ranking tasks, a dilemma is often encountered with classical “product rating” methods: A single ranking model that is trained on the entire collection of data from all tasks may fail to capture the peculiarity of each ranking problem. However, training each model per task using the task-specific data ignores the potential commonalities among related tasks. As an example, consider the problem of recommending movies to multiple users. The recommendation should rely on the user’s own data, e.g., the user profile and the history of movies that were enjoyed previously. Yet it is also reasonable to consider similar users’ preferences. It is thus desirable to combine individual traits and group characteristics to obtain a result that is greater than the sum of its parts.

The problem of jointly solving several related learning tasks by leveraging the commonalities among the tasks has been studied in the machine learning community as multitask learning. Empirical findings have demonstrated the advantages of multitask learning over single-task learning across a variety of application domains [3,16]. Moreover, it has been shown that combining information from several tasks can improve the performance when training data for each task are relatively scarce. Existing works in multitask learning concentrate on solving classification problems for which the objective is to assign instances to one of the several non-ordered categories. The ranking problem that we study resembles classification as both assign one of the several possible labels to an instance; however, it differs in that there is an order relation among the labels. To the best of our knowledge, there have been few attempts to apply multitask learning to the problem of ranking instances.

We herein study the problem of learning to rank instances that belong to multiple related tasks. Specifically, we adopt the online learning paradigm for its simplicity and effectiveness [19]. Online learning represents a family of efficient algorithms that can build the predictor incrementally by processing the training data sequentially, in contrast to batch learning algorithms, which train the predictor by learning the entire dataset all at once. It operates on a sequence of data by processing the elements one by one. In each round, the learner receives an input, makes a prediction using an internal hypothesis, which is retained

in memory, and subsequently learns the true label. It uses the new example to modify its hypothesis according to predefined rules. The objective is to minimize the total number of rounds with incorrect predictions. Online learning algorithms are fast and simple and require few statistical assumptions, thereby rendering them suitable for a wide range of applications. They can scale well to large amounts of data and are particularly suitable for real-world applications in which data arrive continuously.

We propose an efficient online ranking method, namely collaborative online ranking, for ranking data that are generated by a group of tasks by combining both group and individual characteristics. The main strategy is to run N online ranking models in parallel for N related tasks. Each model corresponds to a single task and is updated by using the task data and leveraging the information that is provided by fellow tasks. We devise two approaches for modeling the task-shared information: One sets it explicitly as the average value of all models (Sect. 4.1), while the other uses a latent representation that is derived via learning (Sect. 4.3). We demonstrate how to apply the large margin principle (Sect. 4.2) and the optimization approximation technique (Sect. 4.3) to improve the ranking accuracy and efficiency, respectively. The experimental results on a synthetic dataset and two real-world problems show that the proposed algorithms can improve the overall performance in learning to rank multiple correlated tasks.

The remainder of this paper is organized as follows: Section 2 reviews related work. Section 3 formally defines the multitask online ranking problem that we study. Section 4 presents the proposed collaborative online ranking algorithms. Section 5 gives the experimental results and discussion. Section 6 concludes this paper.

2 Related work

Learning-to-rank involves many subproblems, of which the nuts and bolts demand a dedicated review that is beyond the scope of this paper. For a comprehensive survey of approaches to learning-to-rank, one can refer to [24]. As discussed previously, what we study here is a subcategory of learning-to-rank, or informally speaking, the “product rating” problem. Given items to rank, we do not consider the relative orders among them. Instead, we are interested in selecting an element from a set of ordered categories and assigning it to each item. This differs from the “document ranking” problem, in which we are required to rank a set of documents based on a query. Since the objective of document ranking is to produce an ordered list of documents, the relative order between two items must be considered, whereas the exact relevance degree of an item is less important. In addition, document ranking (especially when applied in web search) always involves a large number of ranking candidates and emphasizes the quality of the top positions of the ranked list. Product rating, in contrast, involves a moderate number of ranks and stresses the accurate assignment of a rank label to each item.

From another perspective, the problem that we study can be categorized as the pointwise approach of learning-to-rank. A characteristic of the pointwise approach that differentiates it from two counterparts (the pairwise approach and the listwise approach, see [24] for details) is that it examines each item independently while ignoring the inter-dependency between items. In various pointwise ranking algorithms, ranking has been modeled as regression, multi-class classification and ordinal regression. When applying existing regression or classification methods to ranking, the key step is the conversion between regression/classification results and ranking scores. The representative conversion approach involves treating regression’s continuous output as the relevance degree [9] and converting the classifier’s discrete output into probabilities via a logistic function [21].

Speaking of ordinal regression, its objective is to predict the label y of an input vector \mathbf{x} , where $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathcal{Y} = \{C_1, C_2, \dots, C_k\}$. In contrast to regression, the response variables in ordinal regression are discrete and finite. In contrast to classification, the labels of ordinal regression imply an ordering, namely $C_1 < C_2 < \dots < C_k$. According to a taxonomy of ordinal regression methods [17], there are ordinal binary decomposition approaches that decompose ordinal labels into several binary labels, solve each binary classification problem independently and combine the outputs into a rank label [8,23]. The decomposed ordinal labels can also be handled using a single model (e.g., neural networks [6]) with ordinal target encoding schemes to incorporate ordering information. Correspondingly, the threshold-based approaches estimate a real-valued predictor $f(\mathbf{x})$ and a set of thresholds \mathbf{b} simultaneously via learning. The thresholds are used to partition a real number axis into intervals, each of which are aligned to an ordinal label. The prediction is the index of the interval to which $f(\mathbf{x})$ belongs. Methods that belong to this category include the perceptron ranking (PRank) [10] and its extensions [11,18], SVMs [8,28] and Gaussian processes [7].

PRank [10] is a perceptron-like online learning algorithm with a threshold-based ranking mechanism. The perceptron can date back to the seminal work on artificial neural networks in the 1950s [26]. As an online classification method, the perceptron learns a model \mathbf{w} from a sequence of data (\mathbf{x}_t, y_t) one element at a time. At each learning round, it compares the predicted label $\hat{y}_t = \text{sign}(\mathbf{w}_t \cdot \mathbf{x}_t)$ with the true label $y_t \in \{-1, 1\}$. If $\hat{y}_t \neq y_t$, the perceptron updates the model as $\mathbf{w}_{t+1} = \mathbf{w}_t + y_t \mathbf{x}_t$. The following works resemble the perceptron as both update the model sequentially, but differ in the employment of more sophisticated update strategies such as the maximum margin principle [12,30] and the second-order constraint [5,13,14]. For a detailed review of online learning, refer to [19].

As discussed above, multitask learning solves a group of related machine learning tasks together [3]. It has been extensively studied in the batch learning paradigm, which assumes that all training samples are available prior to the learning process [15,16]. For online multitask learning, however, the tasks must be processed in parallel with data arriving continuously. A class of online multitask learning methods trains one model per task and makes them interact with one another based on a matrix that represents the inter-task relationships. The relationship matrix can be specified in advance [4] or determined by solving an optimization problem in the learning process [27]. Aside from imposing the task relationship explicitly, another common approach is to add regularization terms to the original objective function. For example, it can be assumed that the objective function is a combination of a global model and a task-specific model [22] or that the model vector is composed of two parts, namely a shared part across all tasks and an individual part that corresponds to each task [29].

We have briefly reviewed representative related work. In the following, we shall formally define the multitask online ranking problem that we study and describe the proposed algorithms in detail.

3 Problem setting

In the multitask online ranking scenario, we are given N related ranking problems, which are also known as tasks. It is assumed that the objects under investigation of all tasks are represented by feature vectors that are drawn from a single feature space. At the beginning of learning round t , the algorithm observes a set of N instances, one for each of the ranking problems. The algorithm predicts the rank for each of the instances it has observed using a ranking model. Then, it receives a feedback that indicates the correct rank and may modify the

model, presumably to improve the likelihood of making an accurate prediction in subsequent rounds.

The main strategy of our approach to multitask online ranking is to maintain several ranking models for each of the related tasks simultaneously and to update them by leveraging both the task-specific training data and the shared knowledge across all tasks. We adopt the perceptron ranking (PRank) [10] as the basic update strategy for learning these models. PRank operates on a collection of instance–rank pairs, namely $\{(\mathbf{x}_t, y_t)\}$, for $t = 1, \dots, T$, where $\mathbf{x}_t \in \mathbb{R}^d$ is a d -dimensional feature vector that represents the instance at round t and its corresponding rank y_t is an element from a finite set $\mathcal{Y} = \{1, \dots, k\}$ with a total order relation. The objective is to learn a mapping from instances to rank labels: $\mathbb{R}^d \rightarrow \mathcal{Y}$.

To learn the mapping, PRank maintains a weight vector $\mathbf{w} \in \mathbb{R}^d$ and a vector \mathbf{b} of increasing thresholds $b[1] \leq \dots \leq b[k-1] \leq b[k] = \infty$, which divides a real number line into k segments, one for each possible rank.¹ Given a new instance \mathbf{x} , PRank computes a score as the inner product between \mathbf{x} and \mathbf{w} . Then, it locates the score on the real line and returns the predicted rank as indicated by the thresholds. Formally, the predicted rank is defined as the index of the first (smallest) threshold $b[r]$ for which $\mathbf{w} \cdot \mathbf{x} < b[r]$, i.e., $\hat{y} = \min_{r \in \{1, \dots, k\}} \{r : \mathbf{w} \cdot \mathbf{x} < b[r]\}$. After making a prediction, the true rank $y \in \mathcal{Y}$ is revealed. An error is considered to occur if the true rank and the prediction do not match, namely $y \neq \hat{y}$.

To make a correct prediction, it is required that $\mathbf{w} \cdot \mathbf{x} > b[r]$ for $r = 1, \dots, y-1$ and $\mathbf{w} \cdot \mathbf{x} < b[r]$ for $r = y, \dots, k-1$.² For simplicity, one can introduce an auxiliary vector $l[1], \dots, l[k-1] = (1, \dots, 1, -1, \dots, -1)$ where the maximal index r for which $l[r] = 1$ is $y-1$. In this way, a predicted rank is correct if $l[r](\mathbf{w} \cdot \mathbf{x} - b[r]) > 0$ for all $r = 1, \dots, k-1$. If PRank makes a mistake, there is at least one threshold, indexed by r , for which the value of $\mathbf{w} \cdot \mathbf{x}$ is on the wrong side of $b[r]$, namely $l[r](\mathbf{w} \cdot \mathbf{x} - b[r]) \leq 0$. To correct this mistake, PRank moves the values of $\mathbf{w} \cdot \mathbf{x}$ and $b[r]$ toward each other. To do so, it modifies the values of $b[r]$ for which $l[r](\mathbf{w} \cdot \mathbf{x} - b[r]) \leq 0$ and replaces them with $b[r] - l[r]$. The value of \mathbf{w} is also replaced by $\mathbf{w} + (\sum l[r])\mathbf{x}$, where the sum is calculated over all the indices r for which there is a prediction error, namely $l[r](\mathbf{w} \cdot \mathbf{x} - b[r]) \leq 0$.

4 Collaborative online ranking

4.1 Basic collaborative online ranking

Our first multitask online ranking algorithm averages the parameters of several ranking models into a single combined model to represent the shared knowledge across all tasks. It shares the same settings as PRank, except that it processes several related ranking tasks in parallel. It is assumed that there are N related tasks whose data arrive in sequence and each sequence contains up to T trials. Denote by (\mathbf{x}_t^i, y_t^i) an instance–rank pair that belongs to the i th task at round t . We adopt a weight vector $\mathbf{w}_t^i \in \mathbb{R}^d$ and a set of k thresholds $b^i[1] \leq \dots \leq b^i[k-1] \leq b^i[k] = \infty$ to represent the ranking model of the i th task. Our objective is to learn N ranking rules for each of the N tasks by using each task’s training data, while leveraging the information that is provided by peer tasks.

As the tasks are related, we can assume that there exists common knowledge that is shared among tasks. We use a *global* model with weight vector $\bar{\mathbf{w}}$ to represent such common

¹ We use $b[i]$ to denote the i th element of the vector \mathbf{b} .

² We exclude the k th threshold because $b[k]$ is fixed to infinity. It is clear that $\mathbf{w} \cdot \mathbf{x} < b[k]$.

knowledge. It is set to an equally weighted sum of all individual models' weight vectors:

$$\bar{\mathbf{w}} = \frac{1}{N} \sum_{i \in N} \mathbf{w}^i \quad (1)$$

We need to define the update rule to modify the weight vector at the end of each learning round whenever a prediction error occurs. Our objective is to realize a balance between two factors. On the one hand, the new weight vector \mathbf{w}_{t+1} should be as similar as possible to the previous weight vector \mathbf{w}_t , which encompasses the knowledge of past training samples. On the other hand, we require the updated weight vector to be similar to the global model's weight vector to incorporate common knowledge that is shared among tasks. Specifically, assuming that the i th task makes an error on the round t , to update its weight vector, we first minimize the sum of the deviations of the new weight from the prior weight and the global weight as follows:

$$\min_{\mathbf{w}^i} \frac{\eta_1}{2} \|\mathbf{w}^i - \mathbf{w}_t^i\|^2 + \frac{\eta_2}{2} \|\mathbf{w}^i - \bar{\mathbf{w}}_t\|^2 \quad (2)$$

where α is a parameter that controls the trade-off between the task-specific model \mathbf{w}_t^i and the global model $\bar{\mathbf{w}}_t$.

We then follow the PRank update rule by adding $(\sum_r l[r])\mathbf{x}^i$, where the summation is applied to all indices r that cause a prediction error, namely $l[r](\mathbf{w}^i \cdot \mathbf{x}^i - b[r]) \leq 0$, to the solution of the optimization problem in Eq. (2). This leads to the following update rule for the simple collaborative online ranking algorithm:

$$\mathbf{w}_{t+1}^i = \frac{\eta_1}{\eta_1 + \eta_2} \mathbf{w}_t^i + \frac{\eta_2}{\eta_1 + \eta_2} \bar{\mathbf{w}}_t + \sum_r l[r] \mathbf{x}_t^i \quad (3)$$

The above formulation aims at realizing a balance between the individual model and the global model: Despite its uniqueness, each individual has commonalities with other tasks. It coherently combines the individual model with the global one. If we set $\eta_2 = 0$, the optimization problem reduces to the approach of learning an individual PRank model without engaging the global model; if we set $\eta_1 = 0$, it reduces to the average of all models. Accordingly, we can tune the contribution of each part by setting appropriate parameters.

Algorithm 1 summarizes the proposed basic collaborative online ranking method. We use $t \in [T]$ to denote $t = 1, \dots, T$ in the pseudo-code.

ALGORITHM 1: Basic Collaborative Online Ranking (COR-BS)

- 1: **Input:** a sequence of instances (\mathbf{x}_t^i, y_t^i) , $i \in [N]$, $t \in [T]$, and parameters η_1 and η_2
 - 2: **Output:** \mathbf{w}_T^i and \mathbf{b}_T^i , $i \in [N]$
 - 3: **Initialize:** $\mathbf{w}_0^i = \mathbf{0}$, $b_0^i[j] = 0$, $j \in [k-1]$, $b_0^i[k] = \infty$
 - 4: **for** $t = 1, \dots, T$ **do**
 - 5: Receive an instance (\mathbf{x}_t^i, y_t^i) for task $i \in [N]$
 - 6: Compute the global model $\bar{\mathbf{w}}_t$ as in Eq. (1)
 - 7: Make a prediction as $\hat{y}_t^i = \min_{r \in [k]} \{r : \mathbf{w}_t^i \cdot \mathbf{x}_t^i - b_t^i[r] < 0\}$
 - 8: Build the auxiliary vector as for $r \in [k-1]$, $l[r] = -1$ if $y_t^i \leq r$; $l[r] = 1$, otherwise
 - 9: Update \mathbf{w}_{t+1}^i as in Eq. (3) for $i \in [N]$
 - 10: **end for**
-

4.2 Collaborative online ranking with a large margin

In this section, we present an improved version of the simple collaborative online ranking algorithm that is presented above. This is motivated by the work of Crammer et al. [11] in which a large margin version of PRank is proposed. Recall that PRank makes a prediction by positioning the value of $\mathbf{w} \cdot \mathbf{x}$ in an interval that is defined by thresholding a continuous real number line. According to [11], the difference between the value of $\mathbf{w} \cdot \mathbf{x}$ and the closest threshold plays a similar role to the notion of a margin in classification problems. Even if PRank yields a correct prediction, the value of $\mathbf{w} \cdot \mathbf{x}$ might lie very close to one of the thresholds, thereby leading to a trivial margin. To ensure a sufficiently large margin after updating, it is necessary for the updated rule $(\mathbf{w}_{t+1}, \mathbf{b}_{t+1})$ to satisfy $\min_r \{(\mathbf{w}_{t+1} \cdot \mathbf{x}_{t+1} - b[r]_{t+1})l[r]_t\} \geq \beta$, where β is a positive constant that corresponds to the margin width.

Following this strategy, we can modify the basic collaborative online ranking algorithm by setting the i th task's new weight vector \mathbf{w}_{t+1}^i and threshold vector \mathbf{b}_{t+1}^i to be the solution to the following optimization problem, with the objective of realizing a margin of at least 1. This can be expressed as a constrained optimization problem as follows:

$$\begin{aligned} \min_{\mathbf{w}^i, \mathbf{b}^i} \quad & \frac{\eta_1}{2} \|(\mathbf{w}^i, \mathbf{b}^i) - (\mathbf{w}_t^i, \mathbf{b}_t^i)\|^2 + \frac{\eta_2}{2} \|(\mathbf{w}^i, \mathbf{b}^i) - (\bar{\mathbf{w}}_t, \bar{\mathbf{b}}_t)\|^2 \\ \text{s.t.} \quad & (\mathbf{w}^i \cdot \mathbf{x}_t^i - b^i[r])l[r] \geq 1 \quad \text{for } r = 1, \dots, k-1 \end{aligned} \quad (4)$$

However, it may be too strict to require every update to realize a margin of at least 1. To relax this constraint, we adopt the technique that is used by the soft margin classifier. Our objective is to realize a margin of at least 1 as often as possible. If the algorithm realizes a margin of less than 1, it will suffer a loss that is defined by the following loss function:

$$\ell(\mathbf{w}^i, b^i[r], \mathbf{x}_t^i, l[r]) = \begin{cases} 0 & (\mathbf{w}^i \cdot \mathbf{x}_t^i - b^i[r])l[r] \geq 1 \\ 1 - (\mathbf{w}^i \cdot \mathbf{x}_t^i - b^i[r])l[r] & \text{otherwise} \end{cases}$$

To relax the fixed-margin constraint, we introduce a nonnegative slack variable ξ into the optimization problem in Eq. (4), thereby making the objective function scale quadratically with the slack variable ξ . This leads to the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}^i, \mathbf{b}^i} \quad & \frac{\eta_1}{2} \|(\mathbf{w}^i, \mathbf{b}^i) - (\mathbf{w}_t^i, \mathbf{b}_t^i)\|^2 + \frac{\eta_2}{2} \|(\mathbf{w}^i, \mathbf{b}^i) - (\bar{\mathbf{w}}_t, \bar{\mathbf{b}}_t)\|^2 + C \sum_{r=1}^{k-1} \xi_r^2 \\ \text{s.t.} \quad & \ell(\mathbf{w}^i, b^i[r], \mathbf{x}_t^i, l[r]) \leq \xi_r \quad \text{for } r = 1, \dots, k-1 \end{aligned}$$

The above optimization problem can be solved via the Lagrange multiplier technique. To do so, we first express its Lagrangian function as:

$$\begin{aligned} \mathcal{L} = & \frac{\eta_1}{2} \|(\mathbf{w}^i, \mathbf{b}^i) - (\mathbf{w}_t^i, \mathbf{b}_t^i)\|^2 + \frac{\eta_2}{2} \|(\mathbf{w}^i, \mathbf{b}^i) - (\bar{\mathbf{w}}_t, \bar{\mathbf{b}}_t)\|^2 + C \sum_{r=1}^{k-1} \xi_r^2 \\ & + \sum_{r=1}^{k-1} \tau_r (1 - (\mathbf{w}^i \cdot \mathbf{x}_t^i - b^i[r])l[r] - \xi_r) \end{aligned} \quad (5)$$

where $\tau_r, r = 1, \dots, k-1$ are nonnegative Lagrange multipliers.

ALGORITHM 2: Collaborative Online Ranking with a Large Margin (COR-LM)

- 1: **Input:** a sequence of instances (\mathbf{x}_t^i, y_t^i) , $i \in [N]$, $t \in [T]$, and parameters η_1 , η_2 and C
 - 2: **Output:** \mathbf{w}_T^i and \mathbf{b}_T^i , $i \in [N]$
 - 3: **Initialize:** $\mathbf{w}_0^i = \mathbf{0}$, $b_0^i[j] = 0$, $j \in [k-1]$, $b_0^i[k] = \infty$
 - 4: **for** $t = 1, \dots, T$ **do**
 - 5: Receive an instance (\mathbf{x}_t^i, y_t^i) for task $i \in [N]$
 - 6: Compute the global model $\bar{\mathbf{w}}_t$ as in Eq. (1)
 - 7: Make a prediction as $\hat{y}_t^i = \min_{r \in [k]} \{r : \mathbf{w}_t^i \cdot \mathbf{x}_t^i - b_t^i[r] < 0\}$
 - 8: Build the auxiliary vector as for $r \in [k-1]$, $l[r] = -1$ if $y_t^i \leq r$; $l[r] = 1$, otherwise
 - 9: Compute τ by solving Eq. (7) with a quadratic solver
 - 10: Update \mathbf{w}_{t+1}^i and \mathbf{b}_{t+1}^i as in Eq. (6) for $i \in [N]$
 - 11: **end for**
-

Differentiating Eq. (5) with respect to \mathbf{w}^i , $b^i[r]$ and ξ_r , respectively, and setting the results to zero yields:

$$\mathbf{w}^i = \frac{\eta_1 \mathbf{w}_t^i + \eta_2 \bar{\mathbf{w}}_t + (\sum_r \tau_r l[r]) \mathbf{x}_t^i}{\eta_1 + \eta_2} \quad (6a)$$

$$b^i[r] = \frac{\eta_1 b_t^i[r] + \eta_2 \bar{b}_t[r] - \tau_r l[r]}{\eta_1 + \eta_2} \quad (6b)$$

$$\xi_r = \frac{\tau_r}{2C} \quad (6c)$$

By substituting the values of \mathbf{w}^i , \mathbf{b}^i and ξ_r into the Lagrangian function in Eq. (5), we obtain the following dual problem:

$$\begin{aligned} \min_{\tau} \quad & \frac{1}{2} \|\mathbf{x}_t^i\|^2 (\sum_r \tau_r l_t[r])^2 + \frac{1}{2} \sum_r \tau_r^2 + \frac{\eta_1 + \eta_2}{4C} \sum_r \tau_r^2 \\ & + \sum_r \tau_r (l_t[r] (\eta_1 (\mathbf{w}_t^i \cdot \mathbf{x}_t^i - b_t^i[r]) + \eta_2 (\bar{\mathbf{w}}_t \cdot \mathbf{x}_t^i - \bar{b}_t[r])) - \eta_1 - \eta_2) \quad (7) \\ \text{s.t.} \quad & \tau_r \geq 0 \quad \text{for } r = 1, \dots, k-1 \end{aligned}$$

The above dual problem can be solved by using any optimization toolbox. Finally, Algorithm 2 lists the pseudo-code of this method for collaborative online ranking with a large margin.

4.3 Collaborative online ranking with latent global information

To represent the shared knowledge among tasks, the collaborative online ranking algorithms presented in previous sections take a uniform average of weight vectors as the global model. A more reasonable alternative approach is not to restrict the global model to any form that is specified beforehand, but to determine the form during the learning process. The algorithm presented in this section divides each task's model \mathbf{w}^i into a global model \mathbf{u} and a personalized model \mathbf{v}^i :

$$\mathbf{w}^i = \mathbf{u} + \mathbf{v}^i \quad \text{for } i = 1, \dots, N$$

The global model \mathbf{u} generalizes a common structure from all tasks, and the personalized model \mathbf{v}^i represents the unique characteristics of individual tasks. Both models are learned incrementally via the online learning process.

In addition to providing a more reasonable global model representation, the algorithm presented in this section attempts to maintain a large margin of at least 1. To realize this objective, we introduce an auxiliary function:

$$\phi_r(\mathbf{w}, \mathbf{b}, \mathbf{x}, l[r]) = \max(0, 1 - (\mathbf{w} \cdot \mathbf{x} - b[r])l[r])$$

We set the i th task's new weight vector \mathbf{w}_{t+1}^i and threshold vector \mathbf{b}_{t+1}^i to the solution of the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}^i, \mathbf{b}^i} \quad & \sum_i \frac{1}{2\eta} \|(\mathbf{v}^i, \mathbf{c}^i) - (\mathbf{v}_t^i, \mathbf{c}_t^i)\|^2 + \frac{1}{2\tau} \|(\mathbf{u}, \mathbf{a}) - (\mathbf{u}_t, \mathbf{a}_t)\|^2 + \sum_i \ell_t^i(\mathbf{w}^i, \mathbf{b}^i) \\ \text{s.t.} \quad & \mathbf{w}^i = \mathbf{u} + \mathbf{v}^i, \mathbf{b}^i = \mathbf{a} + \mathbf{c}^i \quad \text{for } i = 1, \dots, N \end{aligned} \quad (8)$$

where $\ell_t^i(\mathbf{w}^i, \mathbf{b}^i)$ is a loss function that is defined as:

$$\ell_t^i(\mathbf{w}^i, \mathbf{b}^i) = \sum_r \phi_r(\mathbf{w}^i, \mathbf{b}^i, \mathbf{x}_t^i, l[r])$$

As discussed previously, the individual task's weight vector \mathbf{w}^i is composed of the global weight \mathbf{u} and the corresponding personalized weight \mathbf{v}^i . The threshold vector \mathbf{b}^i is composed similarly. The first term in Eq. (8) aims at ensuring that the new models are near the old ones, while the second term aims at minimizing the loss of the new models on the current examples. In addition, η and τ are positive parameters for balancing these two terms.

The optimization problem in Eq. (8) can be reformulated as its dual form, which is a standard quadratic programming problem. Although many off-the-shelf toolboxes are available for solving quadratic programming problems, they typically incur a high computational cost, especially when dealing with a large amount of data. To address this problem, we replace the loss function in Eq. (8) with its linear approximation at the current solution:

$$\ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i)(\mathbf{w}^i, \mathbf{b}^i) = \ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) + \mathbf{g}_t^{\mathbf{w}^i} \cdot (\mathbf{w}^i - \mathbf{w}_t^i) + \mathbf{g}_t^{\mathbf{b}^i} \cdot (\mathbf{b}^i - \mathbf{b}_t^i) \quad (9)$$

Terms $\mathbf{g}_t^{\mathbf{w}^i}$ and $\mathbf{g}_t^{\mathbf{b}^i}$ are the derivatives of ℓ_t^i :

$$\mathbf{g}_t^{\mathbf{w}^i} = \nabla_{\mathbf{w}^i} \ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) = \sum_r \text{sign}[\phi_r(\mathbf{w}_t^i, \mathbf{b}_t^i, \mathbf{x}_t^i, y_t^i)](-l[r]\mathbf{x}_t^i) \quad (10a)$$

$$\mathbf{g}_t^{\mathbf{b}^i} = \nabla_{\mathbf{b}^i} \ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) = \sum_r \text{sign}[\phi_r(\mathbf{w}_t^i, \mathbf{b}_t^i, \mathbf{x}_t^i, y_t^i)]l[r]\mathbf{e}_r \quad (10b)$$

where $\mathbf{e}_r \in \mathbb{R}^{k-1}$ is equal to 1 at the r th position and 0 elsewhere.

Substituting Eq. (9) into Eq. (8), we can reformulate the original optimization problem as:

$$\begin{aligned} \min_{\mathbf{w}^i, \mathbf{b}^i} \quad & \sum_i \left[\frac{1}{2\eta} \|(\mathbf{v}^i, \mathbf{c}^i) - (\mathbf{v}_t^i, \mathbf{c}_t^i)\|^2 + \mathbf{g}_t^{\mathbf{w}^i} \cdot (\mathbf{w}^i - \mathbf{w}_t^i) + \mathbf{g}_t^{\mathbf{b}^i} \cdot (\mathbf{b}^i - \mathbf{b}_t^i) \right] \\ & + \frac{1}{2\tau} \|(\mathbf{u}, \mathbf{a}) - (\mathbf{u}_t, \mathbf{a}_t)\|^2 \end{aligned}$$

We omit $\ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i)$ as it does not affect the solution of the problem.

Setting the derivative of the above objective function to zero yields:

$$\begin{aligned} \mathbf{v}_{t+1}^i &= \mathbf{v}_t^i - \eta \mathbf{g}_t^{\mathbf{w}^i} \\ \mathbf{c}_{t+1}^i &= \mathbf{c}_t^i - \eta \mathbf{g}_t^{\mathbf{b}^i} \end{aligned}$$

ALGORITHM 3: Collaborative Online Ranking with Latent Global Information (COR-LT)

- 1: **Input:** a sequence of instances (\mathbf{x}_t^i, y_t^i) , $i \in [N]$, $t \in [T]$, and parameters η and τ
 - 2: **Output:** \mathbf{w}_T^i and \mathbf{b}_T^i , $i \in [N]$
 - 3: **Initialize:** $\mathbf{w}_0^i = \mathbf{0}$, $b_0^i[j] = 0$, $j \in [k-1]$, $b_0^i[k] = \infty$
 - 4: **for** $t = 1, \dots, T$ **do**
 - 5: Receive an instance (\mathbf{x}_t^i, y_t^i) for task $i \in [N]$
 - 6: Make a prediction as $\hat{y}_t^i = \min_{r \in [k]} \{r : \mathbf{w}_t^i \cdot \mathbf{x}_t^i - b_t^i[r] < 0\}$
 - 7: Build the auxiliary vector as for $r \in [k-1]$, $l[r] = -1$ if $y_t^i \leq r$; $l[r] = 1$, otherwise
 - 8: Compute $\mathbf{g}_t^{\mathbf{w}^i}$ and $\mathbf{g}_t^{\mathbf{b}^i}$ as in Eq. (10) for $i \in [N]$
 - 9: Update \mathbf{w}_{t+1}^i and \mathbf{b}_{t+1}^i as in Eq. (11) for $i \in [N]$
 - 10: **end for**
-

$$\begin{aligned}\mathbf{u}_{t+1} &= \mathbf{u}_t - \tau \sum_j \mathbf{g}_t^{\mathbf{w}^j} \\ \mathbf{a}_{t+1} &= \mathbf{a}_t - \tau \sum_j \mathbf{g}_t^{\mathbf{b}^j}\end{aligned}$$

We can obtain the optimal closed-form solution to Eq. (8) as follows:

$$\mathbf{w}_{t+1}^i = \mathbf{u}_{t+1} + \mathbf{v}_{t+1}^i = \mathbf{w}_t^i - \eta \mathbf{g}_t^{\mathbf{w}^i} - \tau \sum_j \mathbf{g}_t^{\mathbf{w}^j} \quad (11a)$$

$$\mathbf{b}_{t+1}^i = \mathbf{a}_{t+1} + \mathbf{c}_{t+1}^i = \mathbf{b}_t^i - \eta \mathbf{g}_t^{\mathbf{b}^i} - \tau \sum_j \mathbf{g}_t^{\mathbf{b}^j} \quad (11b)$$

Updating according to such a closed-form solution is much more efficient than invoking a quadratic programming routine. Finally, Algorithm 3 summarizes our method.

To theoretically evaluate the performance of the proposed algorithm for collaborative online ranking with latent global information, we introduce the definition of regret for an online learning algorithm \mathcal{A} as follows:

$$\text{Regret}(\mathcal{A}) = \sum_{t=1}^T \sum_{i=1}^N \ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) - \min_{\mathbf{w}, \mathbf{b}} \sum_{t=1}^T \sum_{i=1}^N \ell_t^i(\mathbf{w}, \mathbf{b})$$

It measures how much regret the algorithm feels if it knows all the losses prior to learning.

Regarding the regret bound of the algorithm that is presented in this section, we have the following theorem. Its proof is provided in ‘‘Appendix A’’.

Theorem 1 *Suppose there are N related ranking tasks. Denote by $\{(\mathbf{x}_t^i, y_t^i)\}$, where $i \in [N]$ and $t \in [T]$, a sequence of multitask ranking samples and let L be the Lipschitz constant of $\ell_t^i(\mathbf{w}^i, \mathbf{b}^i)$ for all i and t . Then, for any vector \mathbf{u} , $(\mathbf{v}^1, \dots, \mathbf{v}^N)$, \mathbf{a} and $(\mathbf{c}^1, \dots, \mathbf{c}^N)$, if we denote $\mathbf{w}^i = \mathbf{u} + \mathbf{v}^i$ and $\mathbf{b}^i = \mathbf{a} + \mathbf{c}^i$ for all i , the regret of the algorithm that is proposed in Sect. 4.3 compared with using \mathbf{w}^i and \mathbf{b}^i can be bounded by:*

$$\begin{aligned}& \sum_{t=1}^T \sum_{i=1}^N [\ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) - \ell_t^i(\mathbf{w}, \mathbf{b})] \\ & \leq \sum_{i=1}^N \frac{1}{2\eta} \|(\mathbf{v}^i, \mathbf{c}^i)\|^2 + \frac{1}{2\tau} \|(\mathbf{u}, \mathbf{a})\|^2 + \sum_{i=1}^N \frac{\eta}{2} L^2 T + \frac{\tau}{2} N^2 L^2 T\end{aligned}$$

Furthermore, if we set $\eta = \|(\mathbf{v}^i, \mathbf{c}^i)\|/(L\sqrt{T})$ and $\tau = \|(\mathbf{u}, \mathbf{a})\|/(NL\sqrt{T})$, the regret of the proposed algorithm can be bounded by:

$$\sum_{t=1}^T \sum_{i=1}^N [\ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) - \ell_t^i(\mathbf{w}, \mathbf{b})] \leq \left(\sum_{i=1}^N \|(\mathbf{v}^i, \mathbf{c}^i)\| + N\|(\mathbf{u}, \mathbf{a})\| \right) L\sqrt{T}$$

Remark According to this theorem, the regret of the algorithm that is proposed in Sect. 4.3 is on the order of $O(\sqrt{T})$. This order is optimal because the loss function is not strongly convex [1].

5 Experimental results

5.1 Experimental testbed

We evaluate the performance of the proposed algorithms on a synthetic dataset and two real-world datasets. The number of tasks, rank count, sample size and dimensionality of each dataset are summarized in Table 1.

Synthetic dataset The synthetic dataset is generated similarly to that used in [10]. Denote by $\mathbf{x} = (x_1, x_2)$ a point that is drawn uniformly at random from the unit square $[0, 1] \times [0, 1] \in \mathbb{R}^2$. The sample \mathbf{x} is assigned a rank y from the set $\{1, 2, 3, 4, 5\}$ according to the following ranking rule: $y = \max_{r \in \{1, \dots, 5\}} \{r : 10((x_1 - 0.5)(x_2 - 0.5)) + \xi > b[r]\}$, where $\mathbf{b} = (-\text{inf}, -1, -0.1, 0.25, 1)$ and ξ is a normally distributed noise with mean μ and standard deviation σ . By varying μ and σ over modest ranges, we can obtain related ranking tasks.

In the experiment, a total of 20 related tasks are generated by setting 20 linearly spaced values from two specified ranges as μ and σ , namely $\mu = \{-0.5 : 0.05 : 0.5\}$ and $\sigma = \{0.1 : 0.02 : 0.5\}$. Each task contains 10,000 samples. As the problem is not linearly separable, a previous work [10] employed a polynomial kernel $K(x_1, x_2) = ((x_1 \cdot x_2) + 1)^2$. We use a similar approach by expanding the kernel function via a mapping $(x_1, x_2) \rightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$.

EachMovie dataset The EachMovie dataset has been used extensively in collaborative filtering research. It contains ratings entered by 72,916 users on 1628 movies. Each user rated movies using one of the six possible ratings: 0, 0.2, 0.4, 0.6, 0.8 and 1. We subtract 0.5 from each rating to polarize the range to $-0.5, -0.3, -0.1, -0.1, 0.3$ and 0.5 as in [10]. To generate multiple related ranking problems, we select 1813 users who have viewed a least 200 movies. Then, these 1813 viewers are divided into two sets: The first set contains 30 users who have viewed exactly 200 movies, while the second set is composed of 1783 users

Table 1 Statistics of datasets used in the experiment

	Synthetic	EachMovie	Multi-domain sentiment
#Tasks	20	30	13
#Ranks	5	6	4
#Samples per task	10,000	200	4191
#Data dimension	6	1783	27,239

who have viewed more than 200 movies. We select one user from the first set and treat his/her ratings as the target ranks. We use the ratings of all users in the second set who have viewed the same movie as the features. A value of zero is assigned to a movie if it is not rated by a user. Finally, we obtain 30 related tasks, each of which contains 200 samples (movie–rank pairs).

Multi-domain sentiment dataset The dataset constructed by Blitzer et al. [2] has been used in the transfer learning study. It contains reviews of 25 types of products from *Amazon.com*: apparel, books, camera and photograph, DVD, electronics, etc. Each review is accompanied by an integer rating of 0 to 4 stars. By excluding products whose review counts are less than 4191, we obtain 13 products (tasks). Review text is converted to a word vector using the TF-IDF representation. We shrink all tasks to equal size (4191 instances) to guarantee that an instance is available for all tasks in each learning round.

5.2 Benchmark setup

We term the basic collaborative online ranking algorithm presented in Sect. 4.1 as COR-BS, its large margin version—collaborative online ranking with a large margin in Sect. 4.2 as COR-LM, and collaborative online ranking with latent global information in Sect. 4.3 as COR-LT. We compare them with PRank [10] and NoPRank (an abbreviation for Norm-Optimized PRank, a large margin based version of PRank) [11]. We select the values of parameters η_1 , η_2 , η and τ from a small range {0.001, 0.01, 0.1, 1, 10, 100} for COR-BS, COR-LM and COR-LT. We set β , which is the width of the margin, to 1 for NoPRank.

To evaluate the performance of ranking multiple related tasks together, we devise two variants of PRank/NoPRank as follows:

- *PRank/NoPRank-Unique* It employs the PRank/NoPRank algorithm to train a ranking model for each task using only its own data. That is to say, every task is associated with a unique ranking model.
- *PRank/NoPRank-Global* It learns a single ranking model from the data for all task by applying the PRank/NoPRank algorithm. At each learning round, the algorithm receives a training instance from each task and uses that instance to update a single weight vector.

We adopt the *cumulative error rate*, namely the ratio of the number of mistakes made by an online learner to the number of samples received to date, as a metric for comparing algorithms. Despite its extensive use in online learning studies, the cumulative error rate does not consider the ordinal preference of the rank labels. For example, if the true rank is 1, then a predicted rank 2 is preferred over 3, even though they are equally erroneous when measured by the error rate. To qualify such an ordinal preference, we compute for each algorithm the *cumulative rank-loss rate* as the ratio of the absolute difference between the true and predicted ranks to the total number of examples that have been received to date. Formally, the cumulative rank-loss rate at round T is $(1/T) \sum_{t=1}^T (|y_t - \hat{y}_t|)$. For both metrics, a lower value corresponds to higher performance of the algorithm.

5.3 Performance evaluation

The experiment is conducted on a PC with a 2.4 GHz CPU and 8 GB RAM. We shuffle the sample order for each dataset randomly and average all tasks’ performance measures at the last learning round into a single value for ease of comparison. Table 2 reports the mean error rates and the mean rank-loss rates, together with their standard deviations, for various

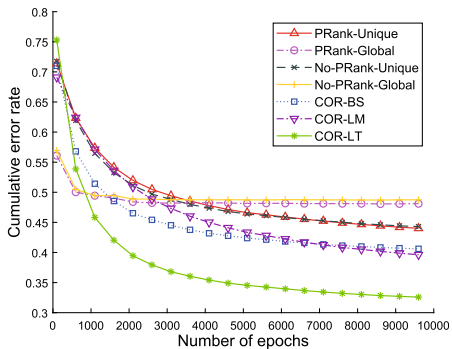
algorithms on the three datasets. Figure 1 depicts the variations of the averaged error rate and the rank-loss rate over the entire online learning process. The runtime for each algorithm, namely the total time that is consumed by updating the models and making predictions, is also listed in Table 2. Our observations from these results are as follows.

The proposed online multitask learning methods (COR-BS, COR-LM and COR-LT) outperform the other three single-task learners in terms of the error rate and the rank-loss rate in most cases. This supports our assumption that learning related tasks collaboratively can outperform models that are trained individually. Among the three datasets, the proposed methods demonstrate the largest performance advantages on the synthetic dataset. This might be because the task correlations are imposed intentionally on the synthetic dataset and, thus, are more explicit and well formed compared to the two real-world datasets, in which tasks are believed to be related to some extent.

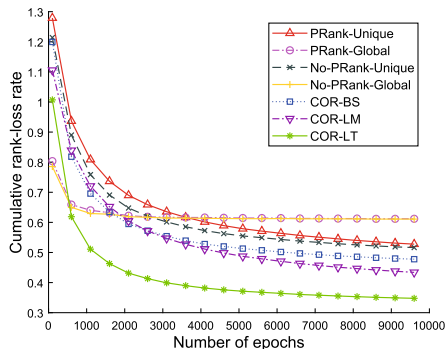
Next, by comparing the three proposed algorithms that employ the collaborative learning strategy (COR-BS, COR-LM and COR-LT), we observe that the latter two, which use the large margin principle and the latent global information representation, tend to produce more accurate results compared to the former simple approach. This indicates that imposing the large margin constraint on the collaborative model has positive effects on solving the

Table 2 Experimental results of the means and standard deviations (in brackets) of all tasks’ “eventual” error rates and rank-loss rates at the last learning round

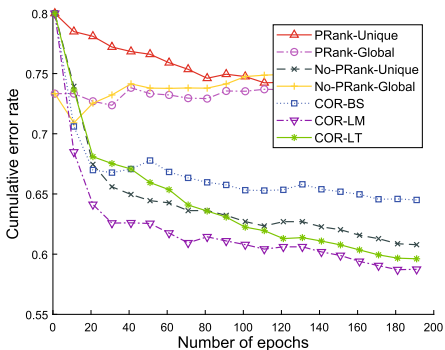
	Error rate	Rank-loss rate	Runtime (s)
(a) Synthetic dataset			
PRank-Unique	0.44 (0.03)	0.52 (0.05)	0.73
PRank-Global	0.48 (0.11)	0.61 (0.27)	1.21
NoPRank-Unique	0.44 (0.03)	0.51 (0.04)	300.31
NoPRank-Global	0.49 (0.10)	0.61 (0.27)	328.78
COR-BS	0.40 (0.04)	0.48 (0.05)	1.98
COR-LM	0.39 (0.04)	0.43 (0.04)	226.96
COR-LT	0.32 (0.03)	0.35 (0.03)	1.49
(b) EachMovie dataset			
PRank-Unique	0.72 (0.11)	1.52 (0.22)	0.68
PRank-Global	0.73 (0.05)	1.36 (0.19)	1.18
NoPRank-Unique	0.61 (0.08)	0.83 (0.17)	13.99
NoPRank-Global	0.75 (0.04)	1.30 (0.16)	16.74
COR-BS	0.64 (0.08)	1.02 (0.19)	1.48
COR-LM	0.59 (0.09)	0.82 (0.18)	12.26
COR-LT	0.59 (0.08)	0.90 (0.15)	1.28
(c) Multi-domain sentiment dataset			
PRank-Unique	0.50 (0.03)	0.77 (0.07)	25.47
PRank-Global	0.48 (0.03)	0.70 (0.05)	27.35
NoPRank-Unique	0.53 (0.03)	0.75 (0.07)	138.74
NoPRank-Global	0.50 (0.03)	0.70 (0.05)	146.31
COR-BS	0.47 (0.03)	0.69 (0.06)	48.92
COR-LM	0.53 (0.03)	0.68 (0.06)	136.28
COR-LT	0.43 (0.03)	0.57 (0.05)	39.17



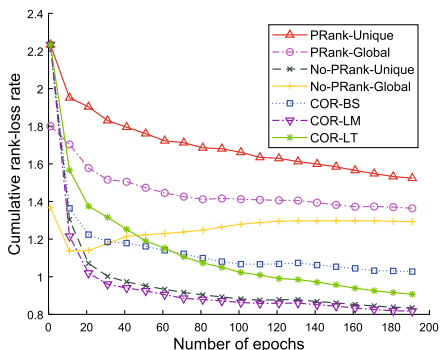
(a) Synthetic — cumulative error rate



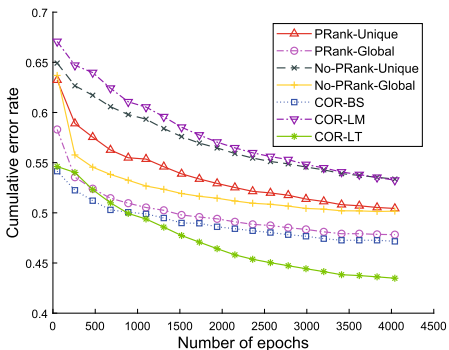
(b) Synthetic — cumulative rank-loss rate



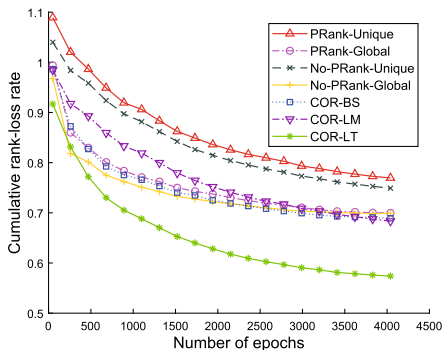
(c) EachMovie — cumulative error rate



(d) EachMovie — cumulative rank-loss rate



(e) Multi-domain sentiment — cumulative error rate



(f) Multi-domain sentiment — cumulative rank-loss rate

Fig. 1 Averaged variations of the cumulative error rate and rank-loss rate over all tasks along the entire online learning process on three datasets

ranking problem. By imposing the large margin constraint, NoPRank always outperforms its counterpart PRank. In addition, by applying both the large margin principle and the multitask learning approach, our COR-LM and COR-LT consistently outperform the large margin single-task learner—NoPRank.

Finally, regarding the computing speed, according to Table 2 COR-LT is considerably faster than NoPRank, although both must solve a quadratic programming problem. NoPRank must invoke an optimization solver in each online learning round. COR-LT accelerates this procedure by approximating the quadratic programming with a closed-form solution. Since both COR-BS and COR-LT have closed-form solutions to optimization problems, they enjoy linear time complexity with respect to the numbers of instances and dimensions. This is not different from typical online learners. Considering the minimal additional time cost that is incurred by COR-BS, COR-LM and COR-LT compared to their counterparts PRank and NoPRank, we conclude that our algorithms are effective and efficient for solving related ranking problems.

6 Conclusions

We study solving multiple related ranking problems together with the objective of using task interactions to improve the rank accuracy. We begin by discussing common ranking problems and highlight that the problem that we study is the assignment of samples into a set of ordered categories, which can be analogous to rating movies based on an ordinal scale. Such a problem can be solved via the learning-to-rank approach, which automatically constructs a ranking model from training data. Through a brief taxonomy of learning-to-rank, we show that our problem is most closely related to ordinal regression, where the objective is to classify patterns using a categorical scale that contains a natural order of the labels. Although it shares similar characteristics with ordinal regression, our problem deals with multiple ranking tasks whose data are available in a sequential order. To the best of our knowledge, few attempts have been made at exploring ranking problems in the online multitask learning setting.

We propose three collaborative online ranking methods that can utilize individual and global models to improve the overall ranking performance for jointly solving multiple related tasks. We present two approaches for modeling the task-shared knowledge: a native approach that takes a uniform average of weight vectors as the global model (Sect. 4.1) and a latent representation approach that learns the global model via training (Sect. 4.3). We show that applying the large margin principle can improve the accuracy of ranking multiple related tasks (Sect. 4.2). We further devise a closed-form approximation of the optimization problem so that we no longer need to invoke a quadratic programming routine repeatedly, thereby rendering the learning process more computational efficient (Sect. 4.3). The experimental results demonstrate that the integration of the global and unique models in the proposed collaborative learning approach leads to performances that are superior to those of either approach alone.

In the future, we wish to extend our experiments to a larger dataset and to additional applications. Our methods assume uniform relations across tasks. However, it is more reasonable to consider the degree of relatedness among tasks. The incorporation of hierarchies and clusters of tasks is also worthy of further study.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant Nos.: 61602356, U1736216, 61571352, U1536202), the Key Research and Development Program of Shaanxi Province, China (Grant No.: 2018GY-002), and the Fundamental Research Funds for the Central Universities of China.

A Appendix

We prove Theorem 1 in Sect. 4.3. First, referring to the optimization problem in Eq. (8), we define:

$$\begin{aligned} \Delta_t &= \sum_{i=1}^N \frac{1}{2\eta} \left[\|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_t^i, \mathbf{c}_t^i\|^2 - \|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_{t+1}^i, \mathbf{c}_{t+1}^i\|^2 \right] \\ &\quad + \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_t, \mathbf{a}_t\|^2 - \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_{t+1}, \mathbf{a}_{t+1}\|^2 \end{aligned} \quad (12)$$

Note that

$$\begin{aligned} \sum_{t=1}^T \Delta_t &= \sum_{t=1}^T \sum_{i=1}^N \frac{1}{2\eta} \left[\|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_t^i, \mathbf{c}_t^i\|^2 - \|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_{t+1}^i, \mathbf{c}_{t+1}^i\|^2 \right] \\ &\quad + \sum_{t=1}^T \left[\frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_t, \mathbf{a}_t\|^2 - \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_{t+1}, \mathbf{a}_{t+1}\|^2 \right] \\ &= \sum_{i=1}^N \frac{1}{2\eta} \left[\|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_1^i, \mathbf{c}_1^i\|^2 - \|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_{T+1}^i, \mathbf{c}_{T+1}^i\|^2 \right] \\ &\quad + \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_1, \mathbf{a}_1\|^2 - \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_{T+1}, \mathbf{a}_{T+1}\|^2 \end{aligned} \quad (13)$$

Combining Eq. (13) with $\mathbf{u}_1 = \mathbf{v}_1^i = 0$, $\mathbf{a}_1 = \mathbf{c}_1^i = 0$ and since $\|\cdot\|^2$ is a nonnegative function, we obtain:

$$\sum_{t=1}^T \Delta_t \leq \sum_{i=1}^N \frac{1}{2\eta} \|\mathbf{v}^i, \mathbf{c}^i\|^2 + \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\|^2 \quad (14)$$

Now, we bound Δ_t from below. Referring to Eq. (12), we obtain:

$$\begin{aligned} \Delta_t &= \sum_{i=1}^N \frac{1}{2\eta} \left[\|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_t^i, \mathbf{c}_t^i\|^2 - \|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_t^i - \eta \mathbf{g}_t^{\mathbf{w}^i}, \mathbf{c}_t^i - \eta \mathbf{g}_t^{\mathbf{b}^i}\|^2 \right] \\ &\quad + \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_t, \mathbf{a}_t\|^2 - \frac{1}{2\tau} \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_t - \tau \sum_j \mathbf{g}_t^{\mathbf{w}^j}, \mathbf{a}_t - \tau \sum_j \mathbf{g}_t^{\mathbf{b}^j}\|^2 \\ &= \sum_{i=1}^N \frac{1}{2\eta} \left[\|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_t^i, \mathbf{c}_t^i\|^2 - \|\mathbf{v}^i, \mathbf{c}^i\| - \|\mathbf{v}_t^i, \mathbf{c}_t^i\|^2 \right. \\ &\quad \left. - 2\eta \mathbf{g}_t^{\mathbf{w}^i} \cdot (\mathbf{v}^i - \mathbf{v}_t^i) - 2\eta \mathbf{g}_t^{\mathbf{b}^i} \cdot (\mathbf{c}^i - \mathbf{c}_t^i) - \|(\eta \mathbf{g}_t^{\mathbf{w}^i}, \eta \mathbf{g}_t^{\mathbf{b}^i})\|^2 \right] \\ &\quad + \frac{1}{2\tau} [\|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_t, \mathbf{a}_t\|^2 - \|\mathbf{u}, \mathbf{a}\| - \|\mathbf{u}_t, \mathbf{a}_t\|^2 \\ &\quad - 2\tau \sum_j \mathbf{g}_t^{\mathbf{w}^j} \cdot (\mathbf{u} - \mathbf{u}_t) - 2\tau \sum_j \mathbf{g}_t^{\mathbf{b}^j} \cdot (\mathbf{a} - \mathbf{a}_t) - \|(\tau \sum_j \mathbf{g}_t^{\mathbf{w}^j}, \tau \sum_j \mathbf{g}_t^{\mathbf{b}^j})\|^2] \end{aligned}$$

$$\begin{aligned}
&= \sum_{i=1}^N \left[\mathbf{g}_t^{\mathbf{w}^i} \cdot (\mathbf{v}_t^i - \mathbf{v}^i) + \mathbf{g}_t^{\mathbf{b}^i} \cdot (\mathbf{c}_t^i - \mathbf{c}^i) - \frac{\eta}{2} \|(\mathbf{g}_t^{\mathbf{w}^i}, \mathbf{g}_t^{\mathbf{b}^i})\|^2 \right] \\
&\quad + \sum_j \mathbf{g}_t^{\mathbf{w}^j} \cdot (\mathbf{u}_t - \mathbf{u}) + \sum_j \mathbf{g}_t^{\mathbf{b}^j} \cdot (\mathbf{a}_t - \mathbf{a}) - \frac{\tau}{2} \|(\sum_j \mathbf{g}_t^{\mathbf{w}^j}, \sum_j \mathbf{g}_t^{\mathbf{b}^j})\|^2
\end{aligned}$$

Since $\mathbf{w}^i = \mathbf{u} + \mathbf{v}^i$, $\mathbf{b}^i = \mathbf{a} + \mathbf{c}^i$, we obtain:

$$\begin{aligned}
\Delta_t &= \sum_{i=1}^N [\mathbf{g}_t^{\mathbf{w}^i} \cdot (\mathbf{w}_t^i - \mathbf{w}^i) + \mathbf{g}_t^{\mathbf{b}^i} \cdot (\mathbf{b}_t^i - \mathbf{b}^i)] - \sum_{i=1}^N \frac{\eta}{2} \|(\mathbf{g}_t^{\mathbf{w}^i}, \mathbf{g}_t^{\mathbf{b}^i})\|^2 \\
&\quad - \frac{\tau}{2} \|(\sum_j \mathbf{g}_t^{\mathbf{w}^j}, \sum_j \mathbf{g}_t^{\mathbf{b}^j})\|^2 \\
&\geq \sum_{i=1}^N [\ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) - \ell_t^i(\mathbf{w}, \mathbf{b})] - \sum_{i=1}^N \frac{\eta}{2} \|(\mathbf{g}_t^{\mathbf{w}^i}, \mathbf{g}_t^{\mathbf{b}^i})\|^2 - \frac{\tau}{2} \|(\sum_j \mathbf{g}_t^{\mathbf{w}^j}, \sum_j \mathbf{g}_t^{\mathbf{b}^j})\|^2 \quad (15)
\end{aligned}$$

Note that for the inequality in Eq. (15), we use the following:

$$\mathbf{g}_t^{\mathbf{w}^i} \cdot (\mathbf{w}_t^i - \mathbf{w}^i) + \mathbf{g}_t^{\mathbf{b}^i} \cdot (\mathbf{b}_t^i - \mathbf{b}^i) \geq \ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) - \ell_t^i(\mathbf{w}, \mathbf{b})$$

This is because $\ell_t^i(\cdot, \cdot)$ is convex.

Therefore, using the upper bound of $\sum_{t=1}^T \Delta_t$ in Eq. (14) and the lower bound of Δ_t in Eq. (15), we obtain:

$$\begin{aligned}
&\sum_{t=1}^T \sum_{i=1}^N [\ell_t^i(\mathbf{w}_t^i, \mathbf{b}_t^i) - \ell_t^i(\mathbf{w}, \mathbf{b})] \\
&\leq \sum_{i=1}^N \frac{1}{2\eta} \|(\mathbf{v}^i, \mathbf{c}^i)\|^2 + \frac{1}{2\tau} \|(\mathbf{u}, \mathbf{a})\|^2 + \sum_{i=1}^N \sum_{t=1}^T \frac{\eta}{2} \|(\mathbf{g}_t^{\mathbf{w}^i}, \mathbf{g}_t^{\mathbf{b}^i})\|^2 \\
&\quad + \frac{\tau}{2} \sum_{t=1}^T \|(\sum_j \mathbf{g}_t^{\mathbf{w}^j}, \sum_j \mathbf{g}_t^{\mathbf{b}^j})\|^2 \\
&\leq \sum_{i=1}^N \frac{1}{2\eta} \|(\mathbf{v}^i, \mathbf{c}^i)\|^2 + \frac{1}{2\tau} \|(\mathbf{u}, \mathbf{a})\|^2 + \sum_{i=1}^N \frac{\eta}{2} L^2 T + \frac{\tau}{2} N^2 L^2 T
\end{aligned}$$

where the last inequality is due to the Lipschitzness of the loss $\ell_t^i(\cdot, \cdot)$.

References

1. Abernethy JD, Bartlett PL, Rakhlin A, Tewari A (2008) Optimal strategies and minimax lower bounds for online convex games. In: 21st Annual conference on learning theory—COLT 2008, Helsinki, Finland, July 9–12, 2008, pp 415–424. <http://colt2008.cs.helsinki.fi/papers/111-Abernethy>
2. Blitzer J, Dredze M, Pereira F (2007) Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: ACL 2007, Proceedings of the 45th annual meeting of the association for computational linguistics, June 23–30, 2007, Prague, Czech Republic. <http://aclweb.org/anthology-new/P/P07/P07-1056.pdf>
3. Caruana R (1997) Multitask learning. Mach Learn 28(1):41–75. <https://doi.org/10.1023/A:1007379606734>

4. Cavallanti G, Cesa-Bianchi N, Gentile C (2010) Linear algorithms for online multitask classification. *J Mach Learn Res* 11:2901–2934. <http://portal.acm.org/citation.cfm?id=1953026>
5. Cesa-Bianchi N, Conconi A, Gentile C (2005) A second-order perceptron algorithm. *SIAM J Comput* 34(3):640–668. <https://doi.org/10.1137/S0097539703432542>
6. Cheng J, Wang Z, Pollastri G (2008) A neural network approach to ordinal regression. In: Proceedings of the international joint conference on neural networks, IJCNN 2008, part of the IEEE world congress on computational intelligence, WCCI 2008, Hong Kong, China, June 1–6, 2008, pp 1279–1284. <https://doi.org/10.1109/IJCNN.2008.4633963>
7. Chu W, Ghahramani Z (2005) Gaussian processes for ordinal regression. *J Mach Learn Res* 6:1019–1041. <http://www.jmlr.org/papers/v6/chu05a.html>
8. Chu W, Keerthi SS (2007) Support vector ordinal regression. *Neural Comput*. 19(3):792–815. <https://doi.org/10.1162/neco.2007.19.3.792>
9. Cossack D, Zhang T (2006) Subset ranking using regression. In: Learning theory, 19th annual conference on learning theory, COLT 2006, Pittsburgh, PA, USA, June 22–25, 2006, Proceedings, pp 605–619. https://doi.org/10.1007/11776420_44
10. Crammer K, Singer Y (2001) Pranking with ranking. In: Advances in neural information processing systems 14 [neural information processing systems: natural and synthetic, NIPS 2001, December 3–8, 2001, Vancouver, British Columbia, Canada], pp 641–647. <http://papers.nips.cc/paper/2023-pranking-with-ranking>
11. Crammer K, Singer Y (2005) Online ranking by projecting. *Neural Comput*. 17(1):145–175. <https://doi.org/10.1162/0899766052530848>
12. Crammer K, Dekel O, Keshet J, Shalev-Shwartz S, Singer Y (2006) Online passive-aggressive algorithms. *J Mach Learn Res* 7:551–585. <http://www.jmlr.org/papers/v7/crammer06a.html>
13. Crammer K, Kulesza A, Dredze M (2009) Adaptive regularization of weight vectors. In: Advances in neural information processing systems 22: 23rd annual conference on neural information processing systems 2009. Proceedings of a meeting held 7–10 December 2009, Vancouver, British Columbia, Canada, pp 414–422. <http://papers.nips.cc/paper/3848-adaptive-regularization-of-weight-vectors>
14. Dredze M, Crammer K, Pereira F (2008) Confidence-weighted linear classification. In: Machine learning, proceedings of the twenty-fifth international conference (ICML 2008), Helsinki, Finland, June 5–9, 2008, pp 264–271. <https://doi.org/10.1145/1390156.1390190>
15. Evgeniou T, Pontil M (2004) Regularized multi-task learning. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining, Seattle, Washington, USA, August 22–25, 2004, pp 109–117. <https://doi.org/10.1145/1014052.1014067>
16. Evgeniou T, Micchelli CA, Pontil M (2005) Learning multiple tasks with kernel methods. *J Mach Learn Res* 6:615–637. <http://www.jmlr.org/papers/v6/evgeniou05a.html>
17. Gutiérrez PA, Pérez-Ortiz M, Sánchez-Monedero J, Fernández-Navarro F, Hervás-Martínez C (2016) Ordinal regression methods: survey and experimental study. *IEEE Trans Knowl Data Eng* 28(1):127–146. <https://doi.org/10.1109/TKDE.2015.2457911>
18. Harrington EF (2003) Online ranking/collaborative filtering using the perceptron algorithm. In: Machine learning, proceedings of the twentieth international conference (ICML 2003), August 21–24, 2003, Washington, DC, USA, pp 250–257. <http://www.aaai.org/Library/ICML/2003/icml03-035.php>
19. Hoi SCH, Sahoo D, Lu J, Zhao P (2018) Online learning: a comprehensive survey. *CoRR*. [arXiv:1802.02871](https://arxiv.org/abs/1802.02871)
20. Li H (2014) Learning to rank for information retrieval and natural language processing, 2nd edn. Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers. <https://doi.org/10.2200/S00607ED2V01Y201410HOLT026>
21. Li P, Burges CJC, Wu Q (2007) Mcrank: learning to rank using multiple classification and gradient boosting. In: Advances in neural information processing systems 20, proceedings of the twenty-first annual conference on neural information processing systems, Vancouver, British Columbia, Canada, December 3–6, 2007, pp 897–904. <http://papers.nips.cc/paper/3270-mcrank-learning-to-rank-using-multiple-classification-and-gradient-boosting>
22. Li G, Hoi SCH, Chang K, Liu W, Jain R (2014) Collaborative online multitask learning. *IEEE Trans Knowl Data Eng* 26(8):1866–1876. <https://doi.org/10.1109/TKDE.2013.139>
23. Lin H, Li L (2012) Reduction from cost-sensitive ordinal ranking to weighted binary classification. *Neural Comput*. 24(5):1329–1367. https://doi.org/10.1162/NECO_a_00265
24. Liu TY (2011) Learning to rank for information retrieval. Springer, Heidelberg
25. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) BPR: Bayesian personalized ranking from implicit feedback. In: UAI 2009, proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, Montreal, QC, Canada, June 18–21, 2009, pp 452–461. https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25

26. Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol Rev* 65(6):386–408
27. Saha A, Rai P, III HD, Venkatasubramanian S (2011) Online learning of multiple tasks and their relationships. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics, AISTATS 2011, Fort Lauderdale, USA, April 11–13, 2011, pp 643–651. <http://www.jmlr.org/proceedings/papers/v15/saha11b/saha11b.pdf>
28. Shashua A, Levin A (2002) Ranking with large margin principle: two approaches. In: Advances in neural information processing systems 15 [neural information processing systems, NIPS 2002, December 9–14, 2002, Vancouver, British Columbia, Canada], pp 937–944. <http://papers.nips.cc/paper/2269-ranking-with-large-margin-principle-two-approaches>
29. Yang H, King I, Lyu MR (2010) Online learning for multi-task feature selection. In: Proceedings of the 19th ACM conference on information and knowledge management, CIKM 2010, Toronto, Ontario, Canada, October 26–30, 2010, pp 1693–1696. <https://doi.org/10.1145/1871437.1871706>
30. Zhao P, Hoi SCH, Jin R (2009) DUOL: a double updating approach for online learning. In: Advances in neural information processing systems 22: 23rd annual conference on neural information processing systems 2009. Proceedings of a meeting held 7–10 December 2009, Vancouver, British Columbia, Canada., pp 2259–2267. <http://papers.nips.cc/paper/3787-duol-a-double-updating-approach-for-online-learning>