Listwise Learning to Rank from Crowds

OU WU and QIANG YOU, Institute of Automation, Chinese Academy of Sciences
FEN XIA, Baidu, Inc.
LEI MA, Institute of Automation, Chinese Academy of Sciences
WEIMING HU, CAS Center for Excellence in Brain Science and Intelligence Technology and Chinese Academy of Sciences

Learning to rank has received great attention in recent years as it plays a crucial role in many applications such as information retrieval and data mining. The existing concept of learning to rank assumes that each training instance is associated with a reliable label. However, in practice, this assumption does not necessarily hold true as it may be infeasible or remarkably expensive to obtain reliable labels for many learning to rank applications. Therefore, a feasible approach is to collect labels from crowds and then learn a ranking function from crowdsourcing labels. This study explores the listwise learning to rank with crowdsourcing labels obtained from multiple annotators, who may be unreliable. A new probabilistic ranking model is first proposed by combining two existing models. Subsequently, a ranking function is trained by proposing a maximum likelihood learning approach, which estimates ground-truth labels and annotator expertise, and trains the ranking function iteratively. In practical crowdsourcing machine learning, valuable side information (e.g., professional grades) about involved annotators is normally attainable. Therefore, this study also investigates learning to rank from crowd labels when side information on the expertise of involved annotators is available. In particular, three basic types of side information are investigated, and corresponding learning algorithms are consequently introduced. Further, the top-k learning to rank from crowdsourcing labels are explored to deal with long training ranking lists. The proposed algorithms are tested on both synthetic and real-world data. Results reveal that the maximum likelihood estimation approach significantly outperforms the average approach and existing crowdsourcing regression methods. The performances of the proposed algorithms are comparable to those of the learning model in consideration reliable labels. The results of the investigation further indicate that side information is helpful in inferring both ranking functions and expertise degrees of annotators.

Categories and Subject Descriptors: D.2.7 [**Software Engineering**]: Distribution and Maintenance—*Documentation*; H.4.0 [**Information Systems Applications**]: General

General Terms: Algorithms

Additional Key Words and Phrases: Listwise learning to rank, crowdsourcing, multiple annotators, probabilistic ranking model, side information

ACM Reference Format:
Ou Wu, Qiang You, Fen Xia, Lei Ma, and Weiming Hu. 2016. Listwise learning to rank from crowds. ACM Trans. Knowl. Discov. Data 11, 1, Article 4 (July 2016), 39 pages.
DOI: http://dx.doi.org/10.1145/2910586

This work is supported by the National Science Foundation of China (NSFC), under grant 61379098.
Authors' addresses: O. Wu, Q. You, and L. Ma, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, 95 Zhongguancun East, Beijing, China; emails: {wuou, yqou, lma}@nlpr.ia.ac.cn; F. Xia, Big Data Laboratory, Baidu, Inc., No. 10 Shangdi 10th Street, Beijing, China; email: fenxia@baidu.com; W. Hu, CAS Center for Excellence in Brain Science and Intelligence Technology, and NLPR, Institute of Automation, Chinese Academy of Sciences, 95 Zhongguancun East, Beijing, China; email: wmhu@nlpr.ia.ac.cn.

DOI: http://dx.doi.org/10.1145/2910586

1. INTRODUCTION

Learning to rank is a relatively new research area which has emerged rapidly in the past decade. It plays a critical role in many applications such as information retrieval, data mining, natural language process, and speech recognition [Cao et al. 2007; Qin et al. 2008]. In a problem related to learning to rank, an instance is a set of objects and a label is a ranking list applied over the objects. In particular, learning to rank aims to construct a ranking function from training instances and ranking labels. In conventional scenario, each label is assumed objective and reliable. This assumption works well and is also used in other conventionally supervised settings such as classification. Many supervised learning studies recently emphasize that producing accurate training labels may be inconceivable or remarkably expensive for many real-world tasks. Alternatively, multiple (possibly subjective or noisy) labels can be provided by annotators with various levels of expert degrees. For example, the Amazon Mechanical Turk (AMT) allows requesters to hire users from all over the world to label data. Any AMT user can opt for the labeling tasks of a user's own choice. In this event, an AMT requester can easily and promptly hire multiple labelers. AMT users, however, are allotted limited control, so acquiring objective and accurate labels is not guaranteed. Thus, learning under multiple annotators must be comprehensively explored.

Substantial research was previously conducted to explore the machine learning methods under multiple annotators. One of the early works [Smyth et al. 1995] that was proposed involved the estimation of the ground truth first and then used the estimated ground truth to learn a model. In 2010, Raykar et al. [2010] presented a probabilistic framework to address classification, regression, and ordinal regression algorithms with multiple annotators. The probabilistic framework was based on a simple yet reasonable assumption, that is, an observed label by an annotator depends on both the true label and expertise degree of the annotator.[1] Their experimental results showed that their framework is superior to the model proposed by Smyth et al. [1995]. Donmez and Garnonell [2010] investigated a case in which the expertise of annotators is time varying as well as developed a sequential Bayesian estimation framework. Yan et al. [2010] introduced new active learning algorithms for learning from crowd labels. Xie et al. [2012] proposed a novel learning framework to assess practical circumstances when annotators refuse to label particular instances and when each annotator is given a different set of instances to label. Other related works focused on considerably different settings [Chen et al. 2010; Yan et al. 2014; Dekel and Shamir 2009].

The above studies paid little attention to learning to rank under a multi-annotator setting. Two recent studies conducted by Volkovs et al. [2012] and Matsui et al. [2014] are similar to the current work, yet significant differences are observed. In particular, the current study focuses on learning to rank, whereas Volkovs et al. [2012] focused on the aggregation of multiple ranking lists and Matsui et al. [2014] paid attention to the estimation of the expertise degrees of involved annotators. Accordingly, the current work adheres to the previous studies by investigating the algorithms for listwise learning to rank involving multiple annotators. Furthermore, credit scores, professional grades, and historical annotation records, which provide valuable side information about the expertise degrees of annotators, may be available in many annotation tasks. For example, in the research of Raykar et al. [2010], the annotators were doctors. Intuitively, the labels made by a doctor with a higher professional grade are presumably more accurate than those made by another lower-grade doctor. Consequently, professional grades can be utilized as side information about the expertise degrees of annotators. In this regard, this study investigates learning algorithms when

[1]Expertise degree is simply a parameter that indicates the accuracy of an annotator's labels over the ground truth labels.

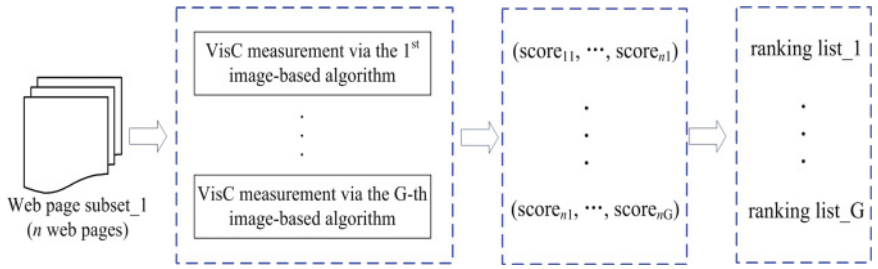

Fig. 1. The simulation of G ranking labels for a training instance (i.e., a web page subset) in web VisC ranking. score_{ij} is the measurement score for the ith web page via the jth image-based algorithm.

side information is available. As far as the researchers know, existing crowdsourcing learning algorithms disregard side information for annotators.

The primary application domain of learning to rank is web information retrieval. This study is also supported by an actual web information retrieval application, that is, visual clutter (VisC) ranking for web pages. The VisCs of web pages determine the accessibility of web pages and are a type of critical factor for accessible web search engines (e.g., Google Accessible Search) [Rosenholtz et al. 2007]. Current VisC measuring algorithms are designed only for images. These algorithms are considered applicable in web VisC ranking if a web page's screenshot is used. However, the computational load becomes significantly high because features are required to be extracted directly on the screenshots of web pages. Consequently, an issue emerges: Can some web page features (e.g., number of texts), which are merely extracted from source code, achieve the same or a comparable performance with that of the state-of-the-art algorithms for images? The answer to this question is considerably important for accessible web search because the time consumption of feature extraction from source code of web pages is significantly lower than that from screenshots. To answer the question, we can resort to an algorithm of learning to rank from crowd labels if the algorithm exists. First, a set of web pages as well as their screenshots is collected. The whole training pages are then divided into several subsets. Three state-of-the-art image-based VisC measuring algorithms are then applied to score the collected web pages on each subset based on their screenshots. The scores for each subset are used to rank the pages in the subset, and each subset then has multiple ranking labels according to the scores obtained by different image-based VisC measuring algorithms. That is, each of the image-based VisC measuring algorithms can be viewed as a simulated annotator. Figure 1 shows the compiling of the construction of the training instances (subsets) and the simulated ranking labels. Once the source-code features are extracted for all the training pages, we can learn a ranking function which takes source-code features as input based on the ranking labels compiled from the multiple simulated annotators. The learning problem is just the procedure of learning to rank from crowdsourcing labels.

This study develops a new learning approach combining labels from multiple annotators and learning a ranking function. In particular, a new probabilistic ranking model is first proposed by combining two classical ones, namely, the Mallow [1957] and the Plackett–Luce (P–L) [Plackett 1975; Luce 1959] models. In this newly proposed ranking model, the ranking function to be learned, ground-truth ranking labels, annotators' expertise degrees, and associated side information on the expertise of annotators can be naturally integrated. Subsequently, specific algorithms for listwise learning to rank from crowds without and with side information are introduced based on the proposed probabilistic ranking model. In each learning algorithm, a maximization likelihood estimation approach is used, and new expectation maximization (EM) [Dempster et al.

1977] procedures are introduced to iteratively infer ground-truth labels, annotators' expertise degrees, and parameters of the ranking function to be learned.

The main contributions of this work can be summarized as follows:

—Unlike the existing studies of learning under multi-annotators that address classification, ordinal regression, and regression, this study focuses on learning to rank. Both learning to rank and labeling under multiple annotations are widely used. Hence, this work can provide more beneficial related applications.
—A new probabilistic ranking model is proposed by combining the Mallow and P–L models. Accordingly, the newly introduced model wholly integrates unsupervised rank aggregation and conventional learning to rank, in which a maximization likelihood optimization framework is used, and new EM procedures are introduced to iteratively infer and learn variables and parameters.
—Existing crowdsourcing learning studies ignore useful side information on the expertise degrees of involved annotators which are usually available in practice. Our early studies [Wu et al. 2011b] did also not explore the side information. Contrarily, this study investigates the types of side information on annotator expertise and proposes corresponding learning algorithms with such information. Experiments on both synthetic and real-world datasets (including two large benchmark learning to rank datasets) suggest that side information improves the performance of the learned ranking function and the accuracy of the estimated annotation expertise degrees.
—The top-k learning to rank setting is considered in this work. In real applications, the length of each ranking list in training is usually long. The standard sampling technique employed in the proposed EM procedure will fail for long ranking lists. Considering that learning to rank with only top-k ground-truth is sufficiently useful, this study investigated the top-k learning to rank from crowds. In our earlier work [Wu et al. 2011b], the top-k setting is not referred.

The rest of the paper is organized into seven sections. Section 2 describes preliminary studies related to our work. Section 3 introduces the proposed listwise learning to rank approach from crowdsourcing labels without considering side information, whereas Section 4 describes the proposed approach with side information. Section 5 extends the proposed approach into the top-k setting. Section 6 reports the experimental results, and Section 7 provides the conclusions of the study.

2. PRELIMINARIES

In this section, some preliminary studies related to our work are briefly introduced. Below are the descriptions of the notations used in the paper.

2.1. Notation and Definitions

Let X be the input space whose elements are instances and each instance is a set of objects. Let Y be the output space whose elements are ranking labels for the instances in X. In this work, to simplify the analysis, the numbers of the objects in each instance are assumed to be identical. Then, an instance $x^{(i)}$ in X is represented by $(x^{(i,1)}, \ldots, x^{(i,N_o)})$, where N_o denotes the number of objects in $x^{(i)}$. Each object is described by N_f-dimensional features and then $x^{(i,j)} \in R^{N_f}$. A ranking label $y^{(i)} \in Y$ for $x^{(i)}$ is denoted by $(y^{(i,1)}, \ldots, y^{(i,N_o)})$, where $y^{(i,j)}$ is the rank assigned to object $x^{(i,j)}$. For convenience, π and σ also depict ranking lists or orderings, where $\pi(j)$ ($\sigma(j)$) is the rank assigned to jth object and $\pi^{-1}(j)$ ($\sigma^{-1}(j)$) is the object index of the jth rank in $\pi(\sigma)$. Let S_{N_o} be a permutation space which contains the set of all possible ranking lists over N_o objects in an instance $x^{(i)}$, and $|S_{N_o}| = N_o!$. For N instances which are independent and identically distributed (i.i.d.) in X and each instance contains N_o objects,

the united permutation space for all possible ranking lists over the objects in these instances is denoted as $S_{N_o}^N$, and $|S_{N_o}^N| = (N_o!)^N$. Let $d : S_{N_o} \times S_{N_o} \rightarrow R$ be a distance function between two ranking lists whose lengths are N_o.

2.2. Probabilistic Ranking Models

A probabilistic ranking model is used to calculate the probability of a given ranking list π given some conditions. This subsection particularly introduces two probabilistic ranking models (i.e., Mallow and P–L) applied in the investigation. The Mallow model is a typical permutation-based model. It is usually used in rank aggregation that ensembles several different ranks into one. The P–L model is a typical score-based model. It is normally used in listwise learning to rank. In the latter section of this paper, these two models are combined as one.

Given the ground-truth ranking π and a dispersion parameter θ of an annotator, the Mallow model calculates the probability of a ranking σ given by the annotator according to

$$p(\sigma|\pi, \theta) = \frac{1}{Z(\pi, \theta)} \exp(\theta \cdot d(\pi, \sigma)), \tag{1}$$

where Z is a normalizing constant:

$$Z(\sigma, \theta) = \sum\nolimits_{\sigma \in S_{N_o}} \exp(\theta \cdot d(\pi, \sigma)). \tag{2}$$

The parameter θ is non-positive; the smaller the value, the more expert the annotator is. When $\theta \rightarrow -\infty$, the probability is 1 if $\pi == \sigma$; and zero, otherwise. When $\theta = 0$, the distribution is uniform, which indicates that the ranking list from the annotator is independent of the ground-truth ranking and can be of any values.

When there are G annotators, let $\mathbf{\Sigma} = (\sigma_1, \ldots, \sigma_G) \in S_{N_o}^G$, where $\sigma_j \ (\in S_{N_o})$ is given by the jth annotator. Given $\mathbf{\Sigma}$ and each annotator's corresponding dispersion parameter θ_j, an extension of the Mallow model is proposed by Lebanon and Lafferty [2002] to calculate the probability of a ground-truth ranking π as follows:

$$p(\pi|\mathbf{\Sigma}, \Theta) = \frac{1}{Z(\mathbf{\Sigma}, \Theta)} p(\pi) \exp\left(\sum_{j=1}^{G} \theta_j \cdot d(\pi, \sigma_j)\right), \tag{3}$$

where the parameter set $\Theta = (\theta_1, \ldots, \theta_G)$ belongs to R^G, and $p(\pi)$ is a prior. The normalizing constant Z is calculated using the following expression:

$$Z(\mathbf{\Sigma}, \Theta) = \sum\nolimits_{\pi \in S_{N_o}} p(\pi) \exp\left(\sum_{j=1}^{G} \theta_j \cdot d(\pi, \sigma_j)\right).$$

The P–L model [Plackett 1975; Luce 1959] calculates the probability of a ranking π given a vector of scores. Different from the case in the Mallow model, the condition in P–L is a vector of scores instead of a ranking and a fixed parameter in Equation (1). The P–L model is parameterized by a score vector $v = (v_1, v_2, \ldots, v_M)$, where $v_i \ (> 0)$ is associated with index i.

$$p(\pi|v) = \prod_{i=1}^{M} \frac{v_{\pi^{-1}(i)}}{v_{\pi^{-1}(i)} + v_{\pi^{-1}(i+1)} + \cdots + v_{\pi^{-1}(M)}}$$

The P–L model has been applied in many machine learning problems [Cheng et al. 2010]. The primary difference between the Mallow and P–L models is that the former

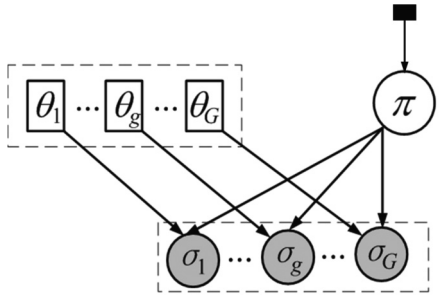

Fig. 2. The graphic model for the generative process defined in Equation (4).

refers to the relationships between rankings, whereas the latter pertains to the relationships between a single ranking and a score vector. Therefore, the Mallow model is usually used in rank aggregation, while the P–L model is usually used in learning to rank. Recently, several new probabilistic ranking models, such as the multinomial preference model, were proposed [Volkovs and Zemel 2014].

2.3. Unsupervised Rank Aggregation

Unsupervised rank aggregation is a fundamental and classical optimization problem addressed in various fields, such as economics, feature selection, and information retrieval [Quin et al. 2010; Lu and Boutilier 2011]. This aggregation combines many different observed ranking lists over a same set of members to infer a "better" ranking list (π) [Soufiani et al. 2013]. Numerous techniques from different research areas have been proposed to address this rank aggregation problem.

Most often, the Mallow model is directly used or extended to aggregate rankings because rank aggregation refers to the direct relationships between rankings. Let $\mathbf{\Sigma} = \{\sigma_1, \ldots, \sigma_G\}$ be the given observed ranking lists and π be the ground-truth ranking list to be inferred. Klementiev et al. [2008] verified that, if the distance is right-invariant,[2] the below generative process is derived based on Equation (3):

$$p(\pi, \mathbf{\Sigma}|\Theta) = p(\pi) \prod_{j=1}^{G} p(\sigma_j|\pi, \theta_j). \tag{4}$$

The generative process is described by Figure 2. First, the ground-truth ranking π is drawn from the prior $p(\pi)$, and σ is subsequently generated by independently drawing $\sigma_1, \ldots, \sigma_G$ from G permutation-based probabilistic models with the same ground-truth ordering π. A maximum-likelihood optimization procedure is used to achieve both the parameters θ_j and to estimate the ground-truth ranking π. This generative process will be used in the succeeding part of our paper.

Volkovs and Zemel [2012, 2014] developed a novel score-based probabilistic ranking model to describe pairwise preferences based on a multinomial generative process. This particular model is extended into supervised settings for rank aggregation.

2.4. Listwise Learning to Rank

Previous learning to rank studies can be divided into three categories, namely, pointwise, pairwise, and listwise [Cao et al. 2007]. A detailed introduction of learning to

[2]For the space X investigated in this study, $\pi * \sigma$ is defined by $\pi * \sigma(i) = \pi(\sigma(i))$. Then if $d(\pi_1, \pi_2) = d(\pi_1 * \sigma, \pi_2 * \sigma)$, the distance d is right-invariant.

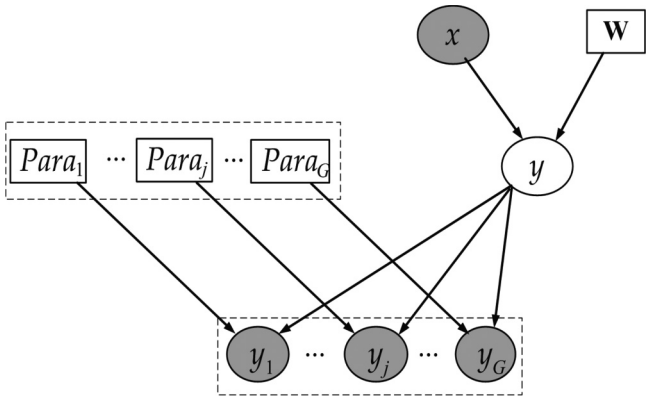

Fig. 3. The underlying graphical model for the learning approach for both classification and regression in Raykar et al. [2010].

rank can be viewed in the work of Liu [2011], which focuses on the listwise approach in particular.

The first procedure in listwise approach learning to rank is defining a loss function over a single truth and predicted orderings. Once the lost function is defined, the parameters of a ranking function are learned according to training loss minimization. The ranking function is generally assumed linear on features:

$$f\left(x^{(i)}\right) = <\mathbf{w}, x^{(i)}> = \mathbf{w}^T x^{(i)}, \tag{5}$$

where $\mathbf{w} \in R^{N_f \times 1}$ and $x^{(i)} \in X$.

Many listwise learning to rank algorithms have been proposed in previous literature [Shen and Li 2011; Geng et al. 2012; Cheng et al. 2013]. Among these algorithms, an effective algorithm, called ListMLE, was proposed by Xia et al. [2008]. ListMLE minimizes the sum of likelihood losses with respect to all training samples. The likelihood loss function is defined as follows:

$$l\left(f\left(x^{(i)}\right), y^{(i)}\right) = -\log P\left(y^{(i)}|x^{(i)}, \mathbf{w}\right),$$

where $y^{(i)} \in Y$ and $P(y^{(i)}|x^{(i)}, \mathbf{w})$ is calculated with the P–L model:

$$P\left(y^{(i)}|x^{(i)}, \mathbf{w}\right) = \prod_{i=1}^{|x^{(i)}|} \frac{\exp\left(\mathbf{w}^T x^{(i,y^{-1}(i))}\right)}{\sum_{k=i}^{|x^{(i)}|} \exp\left(\mathbf{w}^T x^{(i,y^{-1}(k))}\right)},$$

where $|x^{(i)}|$ is the number of objects in the instance $x^{(i)}$. ListMLE utilizes the stochastic gradient descent as the algorithm in searching for (local) optimal parameter $\mathbf{w}$. The detailed procedures of this pursuit can be determined in the research of Xia et al. [2008].

2.5. Learning for Classification and Regression from Crowds

As presented earlier, Raykar et al. [2010] presented a classical probabilistic approach to address learning for classification and regression from multiple annotators. In both classification and regression, the probabilistic framework is based on the underlying graphical model as shown in Figure 3. In the learning task for classification, the value of y in Figure 3 is a categorical label; in the learning task for regression, the value of y in Figure 3 is a real number. The variable $Para_j$ describes the expertise degree of the jth annotator who produces the label y_j. The parameter vector $\mathbf{w}$ represents the classifier or regression function to be learnt.

In this study, the graphical model depicted in Figure 3 is followed and extended to characterize the relationships among objects' features, ground-truth rankings, human annotations, and human expertise degrees. However, there are two distinct differences between this study and Raykar et al.'s study. First, the whole learning tasks are different. This study investigates learning to rank from crowdsourcing labels, whereas Raykar et al.'s study investigated learning for classification, regression, and ordinal regression from crowdsourcing labels. Secondly, the side information for annotators' expertise degrees is considered and utilized in this study, whereas almost all existing studies (including Raykar et al.'s study) ignore the side information in learning from crowdsourcing.

2.6. Learning with Side Information

Side information is attainable and useful in numerous learning tasks, such as classification [Chen et al. 2012; Xu et al. 2013], clustering [Aggarwal et al. 2012; Zhao and Yu 2013], and metric learning [Xing et al. 2003; Wu et al. 2011a]. Given that side information is supplementary to standard supervised information (e.g., labels), learning performances can be substantially improved when the former is effectively utilized. The types of side information used in existing studies can be roughly divided into the following two types.

—*Range side information*. In classification, the goal of learning is usually to train a feature weight vector $\mathbf{w}$. In some classification tasks such as text categorization, it is feasible for domain experts to encode knowledge for the sign of the weights for some features which are referred to as labeled features [Liu et al. 2004]. For example, if the sign of $\mathbf{w}_i$ is judged to be positive, then the side information can be represented by $\mathbf{w}_i > 0$.

—*Relationship side information*. Small et al. [2011] investigated the learning problems when the importance ordering for some features has been given by domain experts. Alternatively, let $\mathbf{w}_i$ and $\mathbf{w}_j$ be the weights corresponding to two features. The side information used by Small et al. [2011] may be $\mathbf{w}_i > \mathbf{w}_j$, which describes the relationships among feature weights. Xing et al. [2003] investigated metric learning with user-provided side information (i.e., similar and dissimilar pairs of data points), which is another form of relationship side information.

In this study, the side information about the expertise degrees of involved annotators is investigated. As no previous studies exist, the types of side information investigated in existing crowdsourcing machine learning will be referred. Besides the two types of side information listed above, a new type will be utilized in our study.

3. LEARNING WITHOUT SIDE INFORMATION ON ANNOTATORS' EXPERTISE

This work considers a typical crowdsourcing labeling strategy, that is, each training instance is labeled by G annotators. In crowdsourcing learning to rank, the training set is accordingly denoted as $D = \{(x^{(i)}, y_1^{(i)}, \ldots, y_G^{(i)})\}_{i=1}^N$, which contains N independently and identically distributed (i.i.d) samples. In particular, this study aims to learn an effective ranking function parameterized by $\mathbf{w}$ (in Equation (4)) and to infer the expertise degrees $\Theta = \{\theta_1, \ldots, \theta_G\}$. The learned ranking function can order the objects for a given test instance, while the inferred expertise degrees can be used to assess the annotators.

Such objective can be achieved by intuitively using a direct two-stage strategy. First, the Mallow model is applied to fuse the ranking labels provided by the annotators to estimate the ground truth and expertise degrees of annotators [Klementiev et al. 2008b]. Conventional learning to rank algorithm (e.g., ListMLE) is then used to train

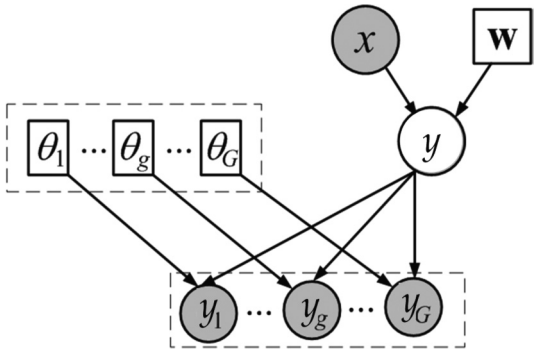

Fig. 4. The proposed probabilistic ranking model.

a ranking function based on the training instances and their estimated ground-truth labels. This direct approach is referred to in the paper as LTRMA-D for brevity. This direct approach, however, is not robust because the second stage fails if the estimated ground truth in the first stage is not good. Consequently, these two stages must be integrated as one. To this end, a new probabilistic ranking model is proposed illuminated by the study conducted by Raykar et al. [2010].

3.1. A New Probabilistic Ranking Model

Based on the generative process in Figure 2 and the graphical model in Figure 3, a new probabilistic ranking model for learning to rank from crowds can be described as shown in Figure 4. In this model, the parameter vector $\mathbf{w} \in R^{N_f \times 1}$ represents the ranking function to be learnt; the parameter set Θ ($\{\theta_1, \ldots, \theta_G\}$) characterizes the expertise degrees of annotators. The variable y is the ground-truth ranking label of an instance x; the variables $\{y_1, \ldots, y_G\}$ are the ranking labels for x from the G annotators. In training, the values of all training instances (x) and their associated crowd labels ($\{y_1, \ldots, y_G\}$) by involved annotators are given.

In Figure 4, the ground-truth ranking label y is conditioned by $v\,(= \langle \mathbf{w}, x \rangle)$. The value of y is now drawn from prior $p(y|v)$. Consequently, the below expression is achieved

$$p(y, y_1, \ldots, y_G|\Theta, v) = p(y|v)p(y_1, \ldots, y_G|y, \Theta) = p(y|x, \mathbf{w}) \prod_{j=1}^{G} p(y_j|y, \theta_j). \tag{6}$$

In the right side of the above equation, $p(y|x, \mathbf{w})$ relies on a score-based probabilistic ranking model, while $p(y_j|y, \theta_j)$ relies on a permutation-based probabilistic ranking model. To our knowledge, the aforementioned P–L model is the most widely used score-based probabilistic ranking model for both full-ordering and partial-ordering rankings; the aforementioned Mallow model is also the most widely used permutation-based probabilistic ranking model for both full-ordering and partial-ordering rankings. Therefore, these two probabilistic ranking models are utilized in the current study. Although both y and y_j are assumed to be full-ordering in the current study, the proposed approach can be easily extended to the learning settings when y and y_j are partial-ordering because the two probabilistic ranking models can also model partial-ordering rankings. In our future work, we also plan to investigate the learning approach when the crowdsourcing labels are pairwise orderings. In this case, new probabilistic ranking models such as the multinomial preference model [Volkovs and Zemel 2014] will be used.

In our training data, a ground-truth label $y^{(i)}$ is assumed to exist (yet unobserved) for the training instance $x^{(i)}$. Consequently, the relationship between $y^{(i)}$ and the label given by the jth annotator ($y_j^{(i)}$) is as follows:

$$p\left(y_j^{(i)}|y^{(i)},\theta_j\right)=\frac{1}{Z(\theta_j)}\exp\left(\theta_j\cdot d\left(y^{(i)},y_j^{(i)}\right)\right). \tag{7}$$

Meanwhile, the relationship between $y^{(i)}$ and the training instance $x^{(i)}$ is constructed by the P–L model as follows:

$$P\left(y^{(i)}|x^{(i)},\mathbf{w}\right)=\prod_{k=1}^{|x^{(i)}|}\frac{\exp\left(\mathbf{w}^T x^{(i,[y^{(i)}(k)]^{-1})}\right)}{\sum_{l=k}^{|x^{(i)}|}\exp\left(\mathbf{w}^T x^{(i,[y^{(i)}(l)]^{-1})}\right)}. \tag{8}$$

The two probabilistic models in Equations (7) and (8) establish the relationships among the observation D ($\{(x^{(i)},y_1^{(i)},\ldots,y_G^{(i)})\}_{i=1}^N$) and the parameters $\mathbf{w}$ and Θ. The succeeding subsection introduces how to estimate the optimal values of the parameters $\mathbf{w}$ and Θ as well as the value of the hidden variable $y^{(i)}$ given D based on a maximum-likelihood estimation procedure.

3.2. A Maximum-Likelihood Estimation Approach

Let $\Omega=(\Theta,\mathbf{w})$ be the parameter set. Given observation D, the likelihood function of Ω can now be factored into the below expression based on Equation (6)

$$\begin{aligned}\Pr(D;\Omega) &= \prod_{i=1}^{N}\Pr\left(x^{(i)},y_1^{(i)},\ldots,y_G^{(i)};\Omega\right)\\ &= \prod_{i=1}^{N}\Pr\left(y_1^{(i)},\ldots,y_G^{(i)}|x^{(i)},\Omega\right)\Pr(x^{(i)})\propto\prod_{i=1}^{N}\Pr\left(y_1^{(i)},\ldots,y_G^{(i)}|x^{(i)},\Omega\right)\\ &= \prod_{i=1}^{N}\sum_{y^{(i)}\in\mathcal{S}_{N_o}}\left\{\Pr\left(y_1^{(i)},\ldots,y_G^{(i)}|y^{(i)},\Theta\right)\Pr\left(y^{(i)}|x^{(i)},\mathbf{w}\right)\right\}\\ &= \prod_{i=1}^{N}\sum_{y^{(i)}\in\mathcal{S}_{N_o}}\left\{\prod_{j=1}^{G}\Pr\left(y_j^{(i)}|y^{(i)},\theta_j\right)\Pr\left(y^{(i)}|x^{(i)},\mathbf{w}\right)\right\}.\end{aligned} \tag{9}$$

Both the Mallow ($\Pr(y_j^{(i)}|y^{(i)},\theta_j)$) and P–L ($\Pr(y^{(i)}|x^{(i)},\mathbf{w})$) models defined in Equations (7) and (8) are integrated in Equation (9). The maximum likelihood estimator is subsequently attained by maximizing the log-likelihood, that is,

$$\bar{\Omega}_{ML}=\{\bar{\Theta},\bar{\mathbf{w}}\}=\arg\max_{\Omega}\{\ln\Pr[D;\Omega]\}. \tag{10}$$

For simplicity, this approach is called LTRMA-MLE. The following section introduces the detailed inference and learning procedures of LTRMA-MLE.

3.3. Inference and Learning for LTRMA-MLE

The maximization of the log-likelihood $\ln\Pr[D;\Omega]$ in Equation (10) is difficult. In this work, $\bar{\Omega}_{ML}$ is estimated by leveraging the EM algorithm [Dempster et al. 1977]. In the EM algorithm, the ground-truth labels ($y^{(i)}$) are taken as missing data. The ground-truth labels ($y^{(i)}$) and $\bar{\Omega}_{ML}$ can then be iteratively estimated. The iteration stops until convergence or the maximum number of iterations is reached. Based on Equation (6),

a new log-likelihood is written as follows:

$$\ln \Pr[D, \mathbf{y}; \Omega] \propto \ln \prod_{i=1}^{N} \Pr\left[y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right], \tag{11}$$

where $\mathbf{y} = \{y^{(1)}, \ldots, y^{(N)}\}$.

The probability of this new log-likelihood corresponds to the generative process described by the proposed model in Figure 4. The EM procedure attempts to achieve the optimal parameters $\bar{\Omega}_{ML}$ by iteratively operating the following procedures.

E-step: This proceeding calculates the expectation of the log-likelihood (i.e., Equation (11)) of the observed data D and the true labels ($y^{(i)}$) with respect to the observed data D and the estimated parameter set $\Omega^{(t)}$ obtained from the previous operation

$$\begin{aligned} Q\left(\Omega, \Omega^{(t)}\right) &= E\left(\ln \prod_{i=1}^{N} \Pr\left[y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right] | D, \Omega^{(t)}\right) \\ &= \sum_{(y^{(1)}, \ldots, y^{(N)}) \in S_{N_o}^N} \left\{\ln \prod_{i=1}^{N} \Pr\left[y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right]\right\} \Pr\left(y^{(1)}, \ldots, y^{(N)} | D, \Omega^{(t)}\right) \\ &= \sum_{i=1}^{N} \sum_{y^{(i)} \in S_{N_o}} \Pr\left(y^{(i)} | D, \Omega^{(t)}\right) \ln \Pr\left[y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right]. \end{aligned} \tag{12}$$

M-step: The parameter set Ω is updated by solving the below maximization problem:

$$\Omega^{(t+1)} = \max_{\Omega} Q\left(\Omega, \Omega^{(t)}\right). \tag{13}$$

The above E- and M-steps are iteratively performed and the algorithm stops until convergence or the maximum number of iterations is reached. For the convergence of the above EM algorithm, interesting readers can refer to the convergence properties of the EM algorithm in McLachlan and Krishnan [1996].

The successive sections describe the details of maximization in the M-step.

3.3.1. Inference. This subsection discusses the mechanism of updating the parameters in each M-step based on Equation (13). The log-likelihood exemplified in Equation (11) is depicted as follows:

$$\begin{aligned} &\ln \prod_{i=1}^{N} \Pr\left[y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right] \\ &= \ln \prod_{i=1}^{N} \left\{\Pr\left[y_1^{(i)}, \ldots, y_G^{(i)} | y^{(i)}, \Theta\right] \Pr\left[y^{(i)} | x^{(i)}, \mathbf{w}\right]\right\} \\ &= \ln \prod_{i=1}^{N} \left\{\prod_{j=1}^{G} \Pr\left[y_j^{(i)} | y^{(i)}, \Theta\right] \Pr\left[y^{(i)} | x^{(i)}, \mathbf{w}\right]\right\} \\ &= \sum_{i=1}^{N} \left\{\sum_{j=1}^{G} \ln \Pr\left[y_j^{(i)} | y^{(i)}, \Theta\right] + \ln \Pr\left[y^{(i)} | x^{(i)}, \mathbf{w}\right]\right\} \\ &= \sum_{i=1}^{N} \ln \Pr\left[y^{(i)} | x^{(i)}, \mathbf{w}\right] - N \sum_{j=1}^{G} \ln Z(\theta_j) + \sum_{i=1}^{N} \sum_{j=1}^{G} \theta_j d\left(y^{(i)}, y_j^{(i)}\right). \end{aligned} \tag{14}$$

With Equation (14), Q in Equation (12) can be transformed into the following equation:

$$\begin{aligned} Q(\Omega, \Omega^{(t)}) &= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \ln \prod_{i=1}^{N} \Pr\left[y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right] \right\} \Pr\left(y^{(1)}, \ldots, y^{(N)} | D, \Omega^{(t)}\right) \\ &= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \sum_{i=1}^{N} \ln \Pr\left[y^{(i)} | x^{(i)}, \mathbf{w}\right] - N \sum_{j=1}^{G} \ln Z(\theta_j) + \sum_{i=1}^{N}\sum_{j=1}^{G} \theta_i d\left(y^{(i)}, y_j^{(i)}\right) \right\} \\ &\quad \times \Pr\left(y^{(1)}, \ldots, y^{(N)} | D, \Omega^{(t)}\right). \end{aligned} \tag{15}$$

Accordingly, new Θ and $\mathbf{w}$ can be obtained and Q can be maximized. At first, we have the following lemma.

LEMMA 2. *Given $\Omega^{(t)}$, for any $\mathbf{w}$, the maximization of Q by Θ is attained by $\Theta = (\theta_1, \ldots, \theta_G)$ such that*

$$E_{\theta_j}(d) = \frac{1}{N} \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} d\left(y^{(i)}, y_j^{(i)}\right) \prod_{i=1}^{N} \Pr\left(y^{(i)} | y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right), j = 1, \ldots, G. \tag{16}$$

PROOF. At first, it is easy to obtain that

$$\sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \prod_{i=1}^{N} \Pr\left(y^{(i)} | y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right) = 1.$$

Then

$$\begin{aligned} &\frac{\partial Q(\Omega, \Omega^{(t)})}{\partial \theta_j} \\ &= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \sum_{i=1}^{N}\sum_{j=1}^{G} \theta_i d\left(y^{(i)}, y_j^{(i)}\right) - N \sum_{j=1}^{G} \ln Z(\theta_j) \right\} \\ &\quad \times \prod_{i=1}^{N} \Pr\left(y^{(i)} | y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right) \\ &= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \sum_{i=1}^{N} d\left(y^{(i)}, y_j^{(i)}\right) - N \frac{\partial}{\partial \theta_j} \sum_{j=1}^{G} \ln Z(\theta_j) \right\} \\ &\quad \times \prod_{i=1}^{N} \Pr\left(y^{(i)} | y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right) \\ &= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \sum_{i=1}^{N} d\left(y^{(i)}, y_j^{(i)}\right) - N \frac{\partial}{\partial \theta_j} \ln Z(\theta_j) \right\} \prod_{i=1}^{N} \Pr\left(y^{(i)} | y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right) \end{aligned}$$

$$= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} d\big(y^{(i)}, y_j^{(i)}\big) \prod_{i=1}^{N} \Pr\big(y^{(i)}|y_1^{(i)},\ldots,y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\big)$$

$$- N\frac{\partial}{\partial\theta_j}\ln Z(\theta_j) \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \prod_{i=1}^{N} \Pr\big(y^{(i)}|y_1^{(i)},\ldots,y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\big)$$

$$= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} d\big(y^{(i)}, y_j^{(i)}\big) \prod_{i=1}^{N} \Pr\big(y^{(i)}|y_1^{(i)},\ldots,y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\big) - N\frac{\partial}{\partial\theta_j}\ln Z(\theta_j). \tag{17}$$

Note that according to the definition of $Z(\theta_j)$, its value only depends on the parameter θ_j, then we have

$$\begin{aligned}\frac{\partial}{\partial\theta_j}\ln Z(\theta_j) &= \frac{\partial}{\partial\theta_j}\ln\sum\nolimits_{\pi\in S_{N_o}} \exp(\theta_j\cdot d(\pi,\sigma)) \\ &= \frac{\sum_{\pi\in S_{N_o}} d(\pi,\sigma)\exp(\theta_j\cdot d(\pi,\sigma))}{\sum_{\pi\in S_{N_o}}\exp(\theta_j\cdot d(\pi,\sigma))} \\ &= E_{\theta_j}(d).\end{aligned} \tag{18}$$

For any $\mathbf{w}$, Q in Equation (12) is maximized by Θ, such that

$$\frac{\partial Q\big(\Omega,\Omega^{(t)}\big)}{\partial\theta_j} = 0.$$

With Equations (15) and (16), the below expression is obtained

$$\begin{aligned}E_{\theta_j}(d) &= \frac{1}{N} \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} d\big(y^{(i)}, y_j^{(i)}\big) \\ &\quad\times \prod_{i=1}^{N} \Pr\big(y^{(i)}|y_1^{(i)},\ldots,y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\big), j = 1,\ldots,G. \quad \square\end{aligned}$$

LEMMA 3. *Given $\Omega^{(t)}$, for any Θ, the maximization of Q in Equation (12) by $\mathbf{w}$ equates to the minimization of the following cross-entropy:*

$$CE = -\sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} \ln\Pr\big[y^{(i)}|x^{(i)}, \mathbf{w}\big] \prod_{i=1}^{N} \Pr\big(y^{(i)}|y_1^{(i)},\ldots,y_G^{(i)}, x^{(i)}, \Theta^{(t)}, \mathbf{w}^{(t)}\big). \tag{19}$$

PROOF. Given $\Omega^{(t)}$, for $\mathbf{w}$, the following formula is achieved:

$$\begin{aligned}
Q\left(\Omega, \Omega^{(t)}\right) &= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \sum_{i=1}^{N} \ln \Pr\left[y^{(i)}|x^{(i)}, \mathbf{w}\right] - N \sum_{j=1}^{G} \ln Z(\theta_i) + \sum_{i=1}^{N}\sum_{j=1}^{G} \theta_i d\left(y^{(i)}, y_j^{(i)}\right) \right\} \\
&\quad \times \prod_{i=1}^{N} \Pr\left(y^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right) \\
&= \sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} \ln \Pr\left[y^{(i)}|x^{(i)}, \mathbf{w}\right] \prod_{i=1}^{N} \Pr\left(y^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right) \\
&\quad + f(\mathbf{w}^{(t)}, \Theta^{(t)}, \Theta). \qquad (20)
\end{aligned}$$

Note that $f(\mathbf{w}^{(t)}, \Theta^{(t)}, \Theta)$ in Equation (20) is independent of $\mathbf{w}$. Then for any Θ, the maximization of Q by $\mathbf{w}$ equates to the minimization of

$$CE = -\sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \sum_{i=1}^{N} \ln \Pr\left[y^{(i)}|x^{(i)}, \mathbf{w}\right] \prod_{i=1}^{N} \Pr\left(y^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \Theta^{(t)}, \mathbf{w}^{(t)}\right).$$

Equation (19) can be seen as the expectation of the log-likelihood loss. The ranking function parameter $\mathbf{w}$ is attained by minimizing the expectation of the likelihood loss. □

3.3.2. Detailed Algorithmic Procedures of LTRMA-MLE. At each EM iteration, Ω (Θ and $\mathbf{w}$) are updated by solving Equation (16) and minimizing Equation (19), respectively. In both equations, obtaining an accurate or analytic minimum is intractable as the number of candidate values of $y^{(i)}$ is large. Therefore, the Metropolis sampling method introduced by Klementiev et al. [2008a] is adopted to approximately obtain the value of the right hand side of Equation (16) and the likelihood loss expectation in Equation (19). During sampling, the sampling space is the permutation space over $N_o (N_o = |x^{(i)}|)$ objects. Based on Figure 4 and the properties of the Mallow model, the proportion of the conditional probability of a new candidate sample (σ') to the previous sampled sample (denoted as $\sigma_m^{(i)}$) for $x^{(i)}$ is calculated as follows:

$$\begin{aligned}
\frac{\Pr\left(\sigma'|D, \Omega^{(t)}\right)}{\Pr\left(\sigma_m^{(i)}|D, \Omega^{(t)}\right)} &= \frac{p\left(\sigma'|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)}{p\left(\sigma_m^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)} \\
&= \frac{P\left(y_1^{(i)}, \ldots, y_G^{(i)}, \sigma'|x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)}{P\left(y_1^{(i)}, \ldots, y_G^{(i)}|x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)} \Bigg/ \frac{P\left(y_1^{(i)}, \ldots, y_G^{(i)}, \sigma_m^{(i)}|x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)}{P\left(y_1^{(i)}, \ldots, y_G^{(i)}|x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)} \\
&= \frac{P\left(y_1^{(i)}, \ldots, y_G^{(i)}, \sigma'|x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)}{\sum_\sigma P\left(y_1^{(i)}, \ldots, y_G^{(i)}|\sigma, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)} \Bigg/ \frac{P\left(y_1^{(i)}, \ldots, y_G^{(i)}, \sigma_m^{(i)}|x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)}{\sum_\sigma P\left(y_1^{(i)}, \ldots, y_G^{(i)}|\sigma, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)}\right)} \\
&= \frac{p\left(\sigma'|x^{(i)}, \mathbf{w}^{(t)}\right) \exp\left(\sum_{j=1}^{G} \theta_j^{(t)} d\left(\sigma', y_j^{(i)}\right)\right)}{p\left(\sigma_m^{(i)}|x^{(i)}, \mathbf{w}^{(t)}\right) \exp\left(\sum_{j=1}^{G} \theta_j^{(t)} d\left(\sigma_m^{(i)}, y_j^{(i)}\right)\right)}.
\end{aligned}$$

Assume that for each instance $x^{(i)}$, we obtain a set of sampled permutations (labels) denoted as $SR(i) = \{\sigma_1^{(i)}, \ldots, \sigma_{N_s}^{(i)}\}$, where N_s is the sampling size. The right-hand side

value (RHS) of Equation (16) is approximately calculated as follows:

$$RHS = \frac{1}{N \cdot N_s} \sum_{i=1}^{N} \sum_{m=1}^{N_s} d\left(\sigma_m^{(i)}, y_j^{(i)}\right), j = 1, \ldots, G.$$

The value of Θ is then calculated based on the value of RHS. In Equation (16), $E_{\theta_j}(d)$ is signified as follows:

$$E_{\theta_j}(d) = \frac{N_o e^{\theta_j}}{1 - e^{\theta_j}} - \sum_{l=1}^{N_o} \frac{l e^{l\theta_j}}{1 - e^{l\theta_j}}. \tag{21}$$

This function is monotonous. Therefore, Θ can be achieved by adopting a binary search approach. The proceedings are shown in Algorithm 1.

ALGORITHM 1: Update Θ

Input: D, $\Theta^{(t)}$, $\mathbf{w}^{(t)}$, $m = 1$, N_s, $\sigma_1^{(i)}$, $i = 1, \ldots, N$.
Output: $\Theta^{(t+1)}$.
Steps:

1. For each $\sigma_m^{(i)}, i \in [1, N]$, choose two indices p, q randomly, and exchange the pth and qth elements of $\sigma_m^{(i)}$ to form a new ordering σ'.
2. Calculate $\alpha_i = \frac{P(\sigma'|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)})}{P(\sigma_m^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \mathbf{w}^{(t)}, \Theta^{(t)})}$. For each $i \in [1, N]$, if $\alpha_i > 1$, $\sigma_{m+1}^{(i)} = \sigma'$; else $\sigma_{m+1}^{(i)} = \sigma_m^{(i)}$ with probability $1 - \alpha_i$ and otherwise $\sigma_{m+1}^{(i)} = \sigma'$. If $m < N_s$, $m = m + 1$ and goto 1.
3. Calculate $\beta_j = \frac{1}{N \cdot N_s} \sum_{i=1}^{N} \sum_{m=1}^{N_s} d(\sigma_m^{(i)}, y_j^{(i)}), j = 1, \ldots, G$.
4. Apply binary search to obtain $\theta_j^{(t+1)}$ according to $\frac{N_o e^{\theta_j}}{1 - e^{\theta_j}} - \sum_{l=1}^{N_o} \frac{l e^{l\theta_j}}{1 - e^{l\theta_j}} = \beta_j, j = 1, \ldots, G$.

The process of minimizing Equation (19) is similar to that in determining parameter $\mathbf{w}$ based on the cross-entropy loss presented by Cao et al. [2007]. However, the computational complexity of minimizing Equation (19) is $O(N_o! N)$. Thus, a heuristic, yet efficient solution[3] is alternatively introduced. Equation (19) measures the distance between two conditional distributions. If the maximum values of these distributions are equal, their distance is likely to be quite small. $P(y^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \Theta^{(t)}, \mathbf{w}^{(t)})$ is assumed to attain its maximum value at $\pi_*^{(i)}$. Equation (19) can then be minimized via a transformation to maximize the below function

$$\ln \prod_{i=1}^{N} P\left(\pi_*^{(i)}|x^{(i)}, \mathbf{w}\right).$$

Accordingly, ListMLE[4] can be leveraged to achieve a new $\mathbf{w}$. The steps of LTRMA-MLE are summarized in Algorithm 2.

[3]The solution is similar to the strategy used in the work of Raykar et al. [2010]. The estimated ground truth at each iteration is used to train a prediction model.
[4]In fact, any other linear or non-linear learning to rank algorithms, which aim to minimize the likelihood loss, can be used in the proposed algorithm. In this study, ListMLE is chosen because of its competing performance reported in previous literature.

ALGORITHM 2: Steps of LTRMA-MLE

Input: D, N_s, $\mathbf{w}^{(0)}$ and $\Theta^{(0)}$, τ_1, τ_2, $t = 0$, $MaxT$.
Output: $\mathbf{w}$, Θ.
Steps:

1. Calculate $\Theta^{(t+1)}$ using Algorithm 1.
2. Repeat the sampling steps 1 and 2 in Algorithm 1 to obtain an ordering set for each $x^{(i)}$.
3. Select the maximum elements in the sampling sets for each $x^{(i)}$. These maximum elements are the estimated ground truth orderings for this particular iteration.
4. Update $\mathbf{w}$ using ListMLE with estimated ground truth.
5. If $t > MaxT$, or $||\Theta^{(t)} - \Theta^{(t+1)}|| < \tau_1$ and $||\mathbf{w}^{(t)} - \mathbf{w}^{(t+1)}|| < \tau_2$, return $\mathbf{w}^{(t+1)}$ and $\Theta^{(t+1)}$; else $t = t + 1$, goto 1.

3.4. Computational Complexity Analysis

According to Algorithm 2, LTRMA-MLE involves t ($\leq MaxT$) iterations and each iteration updates both Θ and $\mathbf{w}$. The computational complexity of the update of Θ depends on the sampling size N_s and the binary search. The computational complexity of the binary search is $O(log(N_c))$ if the number of candidates in binary search is set to N_c. Then the computational complexity of the update of Θ is $O((N_s * N + log(N_c)) * G)$, where N is the number of instances and G is the number of annotators. For the update of $\mathbf{w}$ achieved by the ListMLE, assume that the number of SGD iterations is N_l, then the computational complexity is $O(N_l * N * N_f)$, where N_f is the feature dimension of each object. Then the whole computational complexity of LTRMA-MLE is $O[t * ((N_s * N + log(N_c)) * G + N_l * N * N_f)]$ which depends heavily on the value of t, N_s, and N_l[5]. The maximum number of t (i.e., $MaxT$) is set to 200 and in our experiments the number of t is usually smaller than 20. The value N_l is limited in [1, 50] in the experiments. It is relatively difficult to choose the value of N_s because no convergence results are known for the extended Mallow model with arbitrary distance [Klementiev et al. 2008a]. When the number of objects (N_o) in each list is not large, a relatively small N_s (e.g., 500) can be sufficient for sampling. Klementiev et al. [2008a] observed that the sampling converged rapidly with $10N_o$ steps in their experiments. However, when the number of objects in each list becomes large, the value of N_s should be quite large (e.g., $10e6$), which dramatically increases the complexity of LTRMA-MLE. Section 5 will deal with the learning approach when each training instance contains a large number of objects.

4. LEARNING WITH SIDE INFORMATION ON ANNOTATORS' EXPERTISE

The previous section discusses learning from crowds without any side information about the annotators. Learning from crowds is initially motivated by the fact that annotators generally have unequal expertise degrees. In the previous literature, numerous elaborated algorithms are proposed to simultaneously infer the expertise degrees of annotators and learn a prediction (e.g., classification) function. In practice, some clues (professional grades, credit scores, or certifications in annotators' participated tasks) about the expertise degrees of involved annotators may exist and may be available during annotation and learning. For example, AMT keeps historical details of the performances of annotators in previous annotations. Therefore, some useful information (e.g., one annotator usually gives more accurate labels than another) in

[5]The values of N and N_f are subject to the involved training data and cannot be changed.

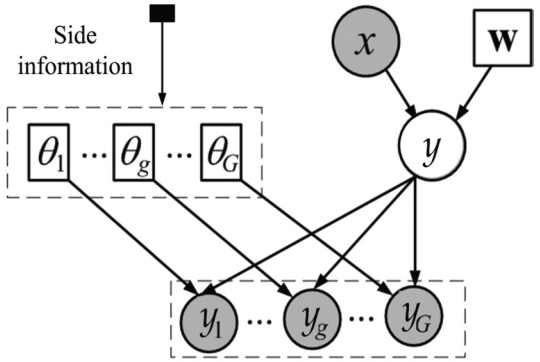

Fig. 5. The proposed probabilistic model in the presence of side information.

a new annotation task in AMT may be available. Some tasks have qualification requirements for the annotators.[6] This useful information, which is normally ignored in existing crowdsourcing machine learning, can also be taken as side information on the expertise degrees of the involved annotators. This section investigates the utilization of side information on annotators in ranking function learning. Some basic types of side information are initially investigated inspired by related studies on side information described in Section 2.6.

4.1. Basic Types of Side Information

In this section, each instance is still considered labeled by G annotators. The jth annotator is associated with the expertise degree θ_j. All side information considered in this work is about the expertise degrees of the annotators as shown in Figure 5. Further, this work does not intend to investigate arbitrary types of side information. Instead, only several basic types of side information are investigated. In the estimation of a set of parameters, the ideal type of side information is that the side information explicitly indicates the exact value(s) of one or more parameters. When this ideal type of side information is unattainable, it is quite useful if the ranges of one or more parameters (i.e., range side information) are given. If both types of side information listed above are unattainable, it is still useful if the relationships between the parameters (i.e., relationship side information) are given. Both the range and relationship types of side information have been widely used in existing studies on leaning with side information. These three basic types of side information can be combined to describe arbitrarily complex side information about the annotators' expertise degrees. Therefore, this study considers these three basic types of side information, which are defined as follows.

—*Type I side information:* In some cases in learning from crowds, the expertise degrees of some annotators are known in advance. Mathematically, this type of side information can be described as follows:

$$\theta_j = v_j, \quad j \in \mathrm{H}, \tag{22}$$

[6]In fact, in the study conducted by Raykar et al. [2010], side information for annotators did exist. The crowdsourcing labels in the experiments conducted by Raykar et al. were from four radiologists. Different radiologists have different professional degrees. The labels from senior radiologists should be more trustworthy than those from junior radiologists. Given that the professional degrees of the four radiologists are easy to obtain, if the professional degrees are different, this type of information can be used to aid the whole learning process.

where ν_j is the expertise degree of the jth annotator, and H is the index set of annotators whose expertise degrees are given. A special case of Type I side information is that the expertise degree of an annotator is approach to $-\infty$. In other words, this annotator can provide ground-truth labels. This special case corresponds to an emerging important topic in crowdsourcing machine learning, that is, learning with crowds and experts [Kajino et al. 2012; Hu et al. 2014]. In conventional crowdsourcing learning, all the labels are assumed to be provided by non-expert annotators and there are no ground-truth labels. Nevertheless, in some applications, it is feasible to invite an expert to provide a small amount of ground-truth labels. In this case, the expertise degree of the expert is approach to $-\infty$.

—*Type II (range) side information:* In some cases in learning from crowds, the ranges of the expertise degrees of some annotators are antecedently provided. Mathematically, this type of side information can be construed as follows:

$$\theta_j \in [\nu_{j1}, \nu_{j2}], \quad j \in \mathrm{H}, \tag{23}$$

where ν_{j1} and ν_{j2} are the range bounds of the expertise degree of the jth annotator, and H is the index set of annotators whose ranges of expertise values are already given. In crowdsourcing learning for classification, it is easy to understand this type of side information. For instance, in a two-category labeling task, if an annotator is judged to be malicious, the expertise degree can be set to be smaller than 0.5; on the contrary, for an annotator with a very high-level expertise degree, the expertise degree of that annotator can be set to be larger than 0.9. In crowdsourcing learning for learning to rank, if an annotator is judged to be malicious before learning, then the expertise degree of the malicious annotator can be set to be that $\theta > 0$[7]. For an annotator with a high-level expertise degree, the annotator's expertise degree can be set to be that $\theta < -3$.

—*Type III (relationship) side information:* In some cases in learning from crowds, the relationships between the expertise degrees of some annotators can be obtained in advance based on information such as annotators' professional grades. Mathematically, this type of side information can be depicted as follows:

$$\theta_j \leq \theta_k, \tag{24}$$

which indicates that the value of the expertise degree for jth annotator is larger than that for kth annotator.[8] In practical learning tasks, for example, if an annotator generated more accurate labels than another specific annotator in all previous labeling tasks, we can denote that the former's expertise degree is larger than the latter's. A special case of Type III side information is that the expertise degree of an annotator is higher than those of other annotators. In other words, the best annotator is known in advance.

Compared with the types of side information utilized in existing studies [Liu et al. 2004; Small et al. 2011; Chen et al. 2012; Xu et al. 2013], this work considers a new type of side information, i.e., Type I side information. The above three basic types of side information conform to practical applications. Nevertheless, no study has been conducted yet on the use of side information on annotators (like the types of side information listed above) in model training. The side information defined above can be perceived as additional constraints of annotators' expertise degrees. In the following subsections, the above three types of side information are integrated in the proposed

[7]Although in the standard Mallow model, θ is defined as non-positive. Nevertheless, in practice, for a malicious annotator, θ can be positive.

[8]In this study, a larger value of θ indicates a lower expertise degree.

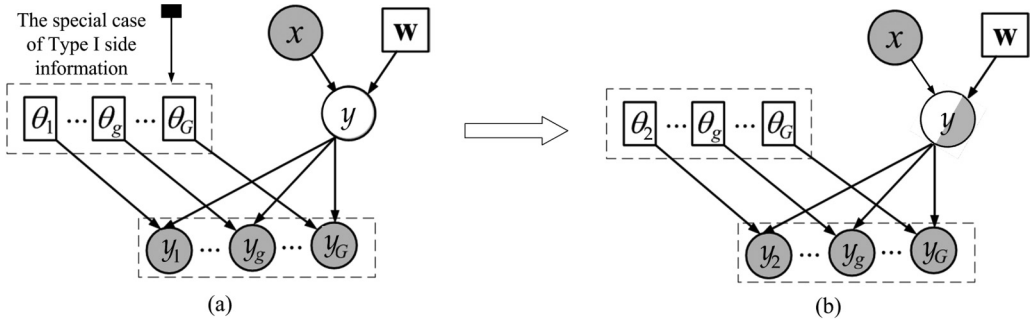

Fig. 6. The probabilistic process for the special case of Type I side information.

maximum likelihood estimation approach, and concrete learning algorithms are proposed for each type of side information.

4.2. Learning with Type I Side Information

Mathematically, in the maximum likelihood estimation approach, learning for the parameter set Ω with Type I side information can be reduced to the below optimization problem

$$\begin{aligned} &\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega} \{\ln \Pr[D; \Omega]\} \\ &s.t. \quad \theta_j = \nu_j, \quad j \in \mathrm{H}. \end{aligned} \tag{25}$$

Without loss of generalization, (25) can be simplified by assuming that only one annotator exists and $\mathrm{H} = \{1\}$. In this case, the learning algorithm is similar to LTRMA-MLE in Section 3 with a slight variation, that is, at each iteration, the expertise degree of the first annotator is simply set to ν_1.

As already mentioned, Type I side information has a special case, that is, when $\nu_1 = -\infty$, the first annotator provides the ground-truth labels for all training instances. Thus, the learning algorithm is reduced to ListMLE. However, this special case contradicts the underlying assumption of the crowdsourcing machine learning that the ground-truth labels are expensive or nearly intractable to be obtained for all training instances. Nevertheless, the special case can be amenable to the case that only a limited number of training instances are with true labels. In previous literature, learning from crowds with partial true labels has been investigated for classification [Kajino et al. 2012; Hu et al. 2014]. The following subsection proposes the learning to rank algorithm from crowds with a limited number of partial true labels.

4.2.1. Learning with Partial True Labels: A Special Case of Type I Side Information. In this scenario, the first annotator provides the true labels for partial training samples. That is, a number of N_P samples are labeled by the first annotator, whereas all instances are labeled by the rest $G-1$ annotators. The training set is denoted as $D = D_P \cup D_U$, where D_P is the subset with true labels and crowd labels, and D_U is the subset with only crowd labels. The subset D_P is represented by $D_P = \{(x^{(i)}, y_1^{(i)}, y_2^{(i)}, \ldots, y_G^{(i)})\}_{i=1}^{N_P}$ where $y_1^{(i)}$ is the ground-truth labels in D_P, and the subset D_U is defined by $D_U = \{(x^{(j)}, y_2^{(j)}, \ldots, y_G^{(j)})\}_{j=Np+1}^{N}$. All N samples in D are independently and identically distributed (i.i.d). This study particularly aims to learn a ranking function parameterized by $\mathbf{w}$ and the expertise degrees Θ of annotators.

A new probabilistic process shown in Figure 6(a) is established and is further simplified, as illustrated in Figure 6(b). In the process specified in Figure 6(b), the ground-truth labels, ys, are partially observed. The maximum likelihood estimator is attained

by maximizing the log-likelihood as follows:

$$\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln \Pr[D;\Omega]\} = \arg\max_{\Omega}\{\ln \Pr[D_P \cup D_U;\Omega]\}. \tag{26}$$

For simplicity, the corresponding algorithm is called LTRMA-S1. The following section introduces the detailed inference and learning steps of LTRMA-S1.

4.2.2. Inference and Learning for LTRMA-S1. From Figure 6(b), the likelihood function of Ω given the observation $D = D_P \cup D_U$ can be factored into the below expression

$$\begin{aligned}
\Pr(D;\Omega) &= \Pr(D_P \cup D_U;\Omega) \\
&\propto \prod_{i=1}^{N_P} \Pr\left(y_1^{(i)}, \ldots, y_G^{(i)}, y^{(i)} | x^{(i)}, \Omega\right) \prod_{j=N_P+1}^{N} \Pr\left(y_2^{(j)}, \ldots, y_G^{(j)} | x_j, \Omega\right) \\
&= \prod_{i=1}^{N_P} \left\{ \Pr\left(y_1^{(i)}, \ldots, y_G^{(i)} | y^{(i)}, \Theta\right) \Pr\left(y^{(i)} | x^{(i)}, \mathbf{w}\right)\right\} \\
&\quad \times \prod_{j=N_P+1}^{N} \sum_{y^{(j)} \in S_{N_o}} \left\{ \Pr\left(y_2^{(j)}, \ldots, y_G^{(j)} | y^{(j)}, \Theta\right) \Pr\left(y^{(j)} | x_j, \mathbf{w}\right)\right\}.
\end{aligned} \tag{27}$$

The EM algorithm is still leveraged, and the truth labels $(y^{(i)})$ in the training subset D_U are taken as missing data. The true labels and $\bar{\Omega}_{ML}$ can then be iteratively estimated. The iteration stops until convergence or the maximum number of iterations is reached.

In E-step, the following equation is first determined:

$$\begin{aligned}
Q(\Omega, \Omega^{(t)}) &= E\left(\ln \prod_{i=1}^{N} \Pr\left[D, y | x^{(i)}, \Omega\right] | D, \Omega^{(t)}\right) \\
&= \ln \prod_{i=1}^{N_P} \Pr\left(y_1^{(i)} | x^{(i)}, \Omega\right) + \sum_{(y^{(2)}, \ldots, y^{(N)}) \in S_{N_o}^{N-1}} \\
&\quad \times \left\{ \ln \prod_{j=N_P+1}^{N} \Pr\left[y_2^{(j)}, \ldots, y_G^{(j)}, y^{(j)} | x_j, \Omega\right] \right\} \Pr\left(y^{(j)} | D, \Omega^{(t)}\right),
\end{aligned} \tag{28}$$

where $\Omega^{(t)}$ is the estimated optimal parameters in the previous iteration.

In M-step, Q is maximized and the parameter set Ω is updated.

Similar to the maximization of Q in Equation (10), we have the following lemmas.

LEMMA 4. *Given $\Omega^{(t)}$, for any given $\mathbf{w}$, Q in Equation (28) is maximized by Θ, such that*

$$\mathrm{E}_{\theta_j}(d) = \frac{1}{N_P} \sum_{i=1}^{N_P} d\left(y_1^{(i)}, y^{(i)}\right) + \sum_{(y^{(N_P+1)}, \ldots, y^{(N)}) \in S_{N_o}^{N-N_P}} \left(\frac{1}{(N - N_P)} \sum_{i=N_P+1}^{N} d\left(y_j^{(i)}, y^{(i)}\right) \right) U(\Omega^{(t)}), \tag{29}$$

where

$$U(\Omega^{(t)}) = \prod_{j=N_P+1}^{N} \Pr\left(y^{(j)} | y_2^{(j)}, \ldots, y_G^{(j)}, x^{(j)}, \Theta^{(t)}, \mathbf{w}^{(t)}\right). \tag{30}$$

LEMMA 5. *Given $\Omega^{(t)}$, for any Θ, the maximization of Q (in Equation (28)) by $\mathbf{w}$ equals to the minimization of the following cross-entropy:*

$$CE = -\ln\left[\prod_{i=1}^{N_P} \mathrm{P}\big(y^{(i)}|x^{(i)}, \mathbf{w}\big)\right] - \sum_{(y^{(N_P+1)},\ldots,y^{(N)})\in S_{N_o}^{N-N_P}} \left\{\ln\left[\prod_{j=N_P+1}^{N} \mathrm{P}\big(y^{(j)}|x^{(j)}, \mathbf{w}\big)\right]\right\} U(\Omega^{(t)}). \tag{31}$$

When $N_P == 0$, Equations (29) and (31) are equal to Equations (16) and (19), respectively; when $N_P == N$, Equation (31) becomes the likelihood loss, and $\mathbf{w}$ can be learned by ListMLE. That is, conventional learning with true labels and crowdsourcing learning with multiple unreliable labels can be conceived as two special cases of the learning problem investigated in this subsection.

4.3. Learning with Type II Side Information

Mathematically, learning the optimal parameter set $\bar{\Omega}_{ML}$ with Type II side information can be reduced to the following optimization problem in the maximum likelihood estimation approach:

$$\begin{aligned} &\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln \Pr[D; \Omega]\} \\ &s.t. \quad \theta_j \in [\nu_{j1}, \nu_{j2}] \quad j \in \mathrm{H}. \end{aligned} \tag{32}$$

That is, the side information becomes the additional constraint for the original optimization problem. In this case, only the expertise degree ranges of some annotators are known. Without loss of generalization, H is assumed {1}. Mathematically, Equation (32) can be rewritten in the following form:

$$\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln \Pr[D; \Omega]\, f(\theta_1)\}, \tag{33}$$

where

$$f(\theta_1) = \begin{cases} \frac{1}{\nu_2 - \nu_1} & \theta_1 \in [\nu_1, \nu_2] \\ 0 & otherwise. \end{cases} \tag{34}$$

As Equation (34) is not differentiable, a new function shown below is used to replace the definition in Equation (34)

$$f_\eta(\theta_1) = \frac{1}{1 + \exp(-\eta(\theta_1 - \nu_1))} - \frac{1}{1 + \exp(-\eta(\theta_1 - \nu_2))} + \epsilon, \tag{35}$$

where $\eta(> 0)$ reflects the confidence of the side information and ϵ is a very small fixed value (e.g., $1e - 10$) which keeps the value of f from zero. Figure 7 shows the curves of $f_\eta(\theta_1)$ when $\eta = 10$ and 100 with $\nu_1 = -3$ and $\nu_2 = -1$.

If $\eta \to 0$, the confidence is zero and $f_\eta(\theta_1)$ becomes ϵ. If $\eta \to +\infty$, the confidence is the highest and the constraint generated from the side information should be absolutely satisfied. The relationship between Equations (34) and (35) is described as follows:

$$f(\theta_1) = \frac{1}{\nu_2 - \nu_1} \lim_{\eta -> +\infty} f_\eta(\theta_1). \tag{36}$$

The factor $\frac{1}{\nu_2 - \nu_1}$ does not affect the final results.

Then, Equation (33) becomes

$$\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln(\Pr[D; \Omega]\, f_\eta(\theta_1))\}. \tag{37}$$

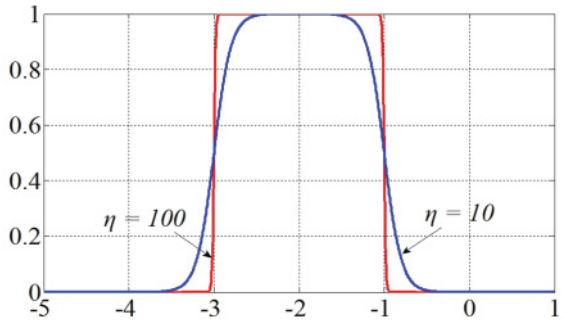

Fig. 7. Two examples of the curve of the function $f_\eta(\theta_1)$.

Note that

$$\frac{\partial f_\eta(\theta_1)}{\partial \theta_1} = \eta\left(1 - \frac{1}{1+\exp(-\eta(\theta_1-\nu_1))} - \frac{1}{1+\exp(-\eta(\theta_1-\nu_2))}\right). \tag{38}$$

We can also define a function $Q(\Omega, \Omega^{(t)})$ similar to Equation (12). With similar inference to the maximization of Equation (12), in the maximization step, the below formula is achieved.

LEMMA 6. *For any* $\mathbf{w}$*, the maximization of* $Q(\Omega, \Omega^{(t)})$ *is attained by* Θ*, such that*

$$\begin{cases} \mathrm{E}_{\theta_1}(d) + \eta\left(1 - \frac{1}{1+\exp(-\eta(\theta_1-\nu_1))} - \frac{1}{1+\exp(-\eta(\theta_1-\nu_2))}\right) = \sum\limits_{(y^{(1)},\ldots,y^{(N)})\in S_{No}^N} \left(\frac{1}{N}\sum\limits_{i=1}^{N} d\big(y^{(i)}, y_1^{(i)}\big)\right) U\big(\Omega^{(t)}\big) \\ \mathrm{E}_{\theta_j}(d) = \sum\limits_{(y^{(1)},\ldots,y^{(N)})\in S_{No}^N} \left(\frac{1}{N}\sum\limits_{i=1}^{N} d\big(y^{(i)}, y_j^{(i)}\}\big)\right) U\big(\Omega^{(t)}\big) \quad j = 2, \ldots, G. \end{cases} \tag{39}$$

If $\theta_1 < \nu_1$, $\eta(1 - 1/1 + \exp(-\eta(\theta_1 - \nu_1)) - 1/1 + \exp(-\eta(\theta_1 - \nu_2))) \approx \eta$*; if* $\theta_1 > \nu_2$, $\eta(1 - 1/1 + \exp(-\eta(\theta_1 - \nu_1)) - 1 + \exp(-\eta(\theta_1 - \nu_2))) \approx -\eta$*. Therefore, when* $\eta \to +\infty$*, the absolute value of the left side of the first equation in Equation (38) is large considering that the value of* θ_1 *is not located within the range of* $[\nu_1, \nu_2]$*. In Equation (38), the values of* θ_j, $j = 2, \ldots, G$ *are achieved by the binary search algorithm. The value of* θ_1 *can be solved with conventional optimizing algorithms such as the Levenberg–Marquardt algorithm [Nocedal and Wright 2006]. In this work, a simple yet efficient method is adopted to achieve the approximately optimal value of* θ_1*. Fifty values are interval extracted from* $[\nu_1, \nu_2]$*. Each of the 50 values are then plugged into the first equation of Equation (38) and the value making the left and right sides the closet is chosen as the value of* θ_1*.*

LEMMA 7. *For any* Θ*, the maximization of* $Q(\Omega, \Omega^{(t)})$ *by* $\mathbf{w}$ *is equivalent to the minimization of the following cross-entropy:*

$$CE = -\sum_{(y^{(1)},\ldots,y^{(N)})\in S_{No}^N} \left\{\ln\left[\prod_{i=1}^{N} \mathrm{P}\big(y^{(i)}|x^{(i)}, \mathbf{w}\big)\right] \times \prod_{i=1}^{N} \mathrm{P}\big(y^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \Theta^{(t)}, \mathbf{w}^{(t)}\big)\right\}. \tag{40}$$

For simplicity, this algorithm is called LTRMA-S2. The procedures of this algorithm are similar to those presented in Algorithms 1 and 2.

4.4. Learning with Type III Side Information

Mathematically, learning the optimal parameter set $\bar{\Omega}_{ML}$ with Type III side information can be reduced to the following optimization problem:

$$\begin{aligned} &\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln \Pr[D; \Omega]\} \\ &s.t. \quad \theta_j \leq \theta_i \quad i, j \in [1, \ldots, G]. \end{aligned} \tag{41}$$

A special case is still considered, in which one annotator has higher expertise degree than all other annotators. Without loss of generalization, the first annotator is assumed to have the highest expertise degree. In this case, Equation (40) is modified as follows:

$$\begin{aligned} &\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln \Pr[D; \Omega]\} \\ &s.t. \quad \theta_1 \leq \theta_j, \quad j \in [2, \ldots, G]. \end{aligned} \tag{42}$$

The constraints in (41) can be rewritten in the following format:

$$f(\Theta) = \begin{cases} 1 \ \ if \quad \theta_1 \leq \theta_j, \quad j = 2, \ldots, G \\ 0 \qquad\qquad otherwise. \end{cases} \tag{43}$$

As Equation (42) is not differentiable, its definition is still replaced by a sigmoid function as follows:

$$f_\eta(\Theta) = \prod_{j=2}^{G} \frac{1}{1 + e^{-\eta(\theta_j - \theta_1)}}, \tag{44}$$

where $\eta(\geq 0)$ reflects the confidence of the side information. If $\eta \to 0$, the confidence is zero and the side information loses effects. If $\eta \to +\infty$, the confidence becomes fairly high and the side information should be completely satisfied. The relationship between Equations (43) and (44) is described as follows:

$$f(\theta_1) = \lim_{\eta -> +\infty} f_\eta(\theta_1). \tag{45}$$

A maximum likelihood estimation approach is also used to estimate the parameters. If η is given, the maximum-likelihood estimator is then attained by maximizing the log-likelihood, that is,

$$\bar{\Omega}_{ML} = \{\bar{\Theta}, \bar{\mathbf{w}}\} = \arg\max_{\Omega}\{\ln(\Pr[D; \Omega]\, f_\eta(\Theta))\}. \tag{46}$$

The EM algorithm is used. Similar to (12), the following expectation is constructed:

$$\begin{aligned} Q(\Omega, \Omega^{(t)}) &= E\left(\prod_{i=1}^{N} \Pr\left(y_1^{(i)}, \ldots, y_G^{(i)} | x^{(i)}, \Omega\right) \Pr(\Omega) | D, \Omega^{(t)}\right) \\ &= E\left(\ln \prod_{i=1}^{N} \prod_{j=1}^{G} \Pr\left(y_j^{(i)} | y^{(i)}, \theta_j\right) \Pr\left(y^{(i)} | x^{(i)}, \mathbf{w}\right) | D, \Omega^{(t)}\right) + \ln f_\eta(\Theta). \end{aligned} \tag{47}$$

Note that

$$\frac{\partial f_\eta(\Theta)}{\partial \theta_1} = -\sum_{j=2,\ldots,G} \frac{\eta e^{\eta(\theta_1 - \theta_j)}}{1 + e^{\eta(\theta_1 - \theta_j)}} \quad \text{and} \quad \frac{\partial f_\eta(\Theta)}{\partial \theta_j} = \frac{\eta e^{\eta(\theta_1 - \theta_j)}}{1 + e^{\eta(\theta_1 - \theta_j)}}, \ j = 2, \ldots, G. \tag{48}$$

Therefore, with an inference procedure similar to the maximization of Equation (10), the following lemmas are obtained.

LEMMA 7. *For any* $\mathbf{w}$*, the maximization of* $Q(\Omega, \Omega^{(t)})$ *is attained by* Θ *such that:*

$$\begin{cases} \mathrm{E}_{\theta_1}(d) - \sum\limits_{j=2,\ldots,G} \frac{\eta e^{\eta(\theta_1-\theta_j)}}{1+e^{\eta(\theta_1-\theta_j)}} = \sum\limits_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left(\frac{1}{N}\sum\limits_{i=1}^{N} d\big(y^{(i)}, y_i^{(1)}\big)\right) U\big(\Omega^{(t)}\big) \\ \mathrm{E}_{\theta_2}(d) + \frac{\eta e^{\eta(\theta_1-\theta_2)}}{1+e^{\eta(\theta_1-\theta_2)}} = \sum\limits_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left(\frac{1}{N}\sum\limits_{i=1}^{N} d\big(y^{(i)}, y_i^{(2)}\big)\right) U\big(\Omega^{(t)}\big) \\ \qquad\qquad \cdots \\ \mathrm{E}_{\theta_G}(d) + \frac{\eta e^{\eta(\theta_1-\theta_G)}}{1+e^{\eta(\theta_1-\theta_G)}} = \sum\limits_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left(\frac{1}{N}\sum\limits_{i=1}^{N} d\big(y^{(i)}, y_i^{(G)}\big)\right) U\big(\Omega^{(t)}\big). \end{cases} \tag{49}$$

Employing conventional optimizing algorithms such as the Levenberg–Marquardt algorithm can solve Equation (49) [Nocedal and Wright 2006].

LEMMA 8. *For any* Θ*, the maximization of* $Q(\Omega, \Omega^{(t)})$ *by* $\mathbf{w}$ *is equivalent to the minimization of the following cross-entropy:*

$$CE = -\sum_{(y^{(1)},\ldots,y^{(N)})\in S_{N_o}^N} \left\{ \ln\left[\prod_{i=1}^{N} \mathrm{P}\big(y^{(i)}|x^{(i)}, \mathbf{w}\big)\right] \times \prod_{i=1}^{N} \mathrm{P}\big(y^{(i)}|y_1^{(i)}, \ldots, y_G^{(i)}, x^{(i)}, \Theta', \mathbf{w}'\big)\right\}. \tag{50}$$

Some observations can be obtained. When $\eta \to 0$*, Equation (49) becomes*

$$\mathrm{E}_{\theta_j}(d) = \sum_{(\pi_1,\ldots,\pi_N)\in S_{N_o}^N} \left(\frac{1}{N}\sum_{i=1}^{N} d\big(\pi_i, y_j^{(1)}\big)\right) U\big(\Theta^{(t)}, \mathbf{w}^{(t)}\big), \quad j = 1, \ldots, G. \tag{51}$$

which is similar to Equation (16). This instance is reasonable because when $\eta \to 0$*, the side information is not meaningful.*

For simplicity, this algorithm is called LTRMA-S3. The proceedings of this algorithm are similar to those in Algorithms 1 and 2.

4.5. Computational Complexity Analysis

The computational complexity of LTRMA-S1 is smaller than that of LTRMA-MLE because some ground-truth values of annotators' expertise degrees are given in LTRMA-S1. Based on the complexity analysis for LTRMA-MLE, the computational complexity of LTRMA-S1 is $O[t * ((N_s * N - N_p + \log(N_c)) * (G-1) + N_l * N * N_f)]$, where t is the number of iterations, N_s is the sampling size, N is the number of instances, N_p is the number of instances with ground-truth labels, N_c is the number of candidate values in binary search, G is the number of annotators, N_l is the number of SGD iterations, and N_f is the feature dimension of the objects in training. The computational complexities of both LTRMA-S2 and LTRMA-S3 differ from that of LTRMA-MLE in the pursuing of the value of Θ at each iteration. In LTRMA-MLE, the value of Θ is achieved by the binary search algorithm which is quite fast. However, in LTRMA-S2, and LTRMA-S3, the value of Θ is achieved by solving the nonlinear equation group defined in Equations (38) and (47). At each iteration of LTRMA-S2, the update for θ_1 can be realized by a simple yet efficient trick if ν_1 and ν_2 are given. We can discretize the range $[\nu_1, \nu_2]$ into 50 folds and then calculate the value of left side of the first equation in Equation (38), the value that is the closet to the right side value is fixed and then the value of θ_1 is obtained. As a consequence, the computation complexity of LTRMA-S2 is $O[t * (N_s * N * G + \log(N_c) * (G-1) + 50 + N_l * N * N_f)]$. For LTRMA-S3, all the values for $\theta_j, j = 1, \ldots, G$ in Equation (47) are coupled and a nonlinear optimization algorithm such as Levenberg–Marquardt should be used. The computational

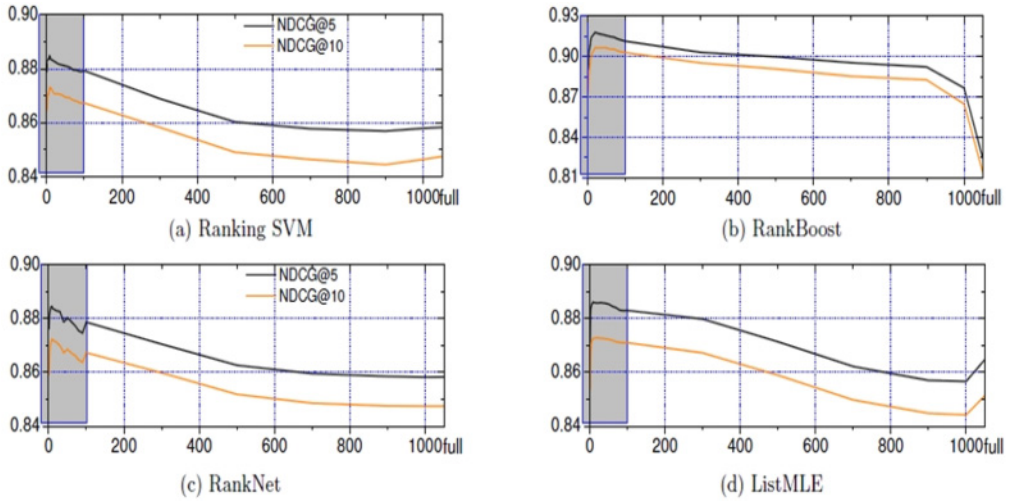

Fig. 8. Performance variations of different ranking algorithms in the top-k setting on a benchmark learning to rank data set (MQ2007) with the increase of k (x-axis) [Lan et al. 2013].

complexity of Levenberg–Marquardt is about $O(G^3)$ [He et al. 2000]. Because G is not large, the Levenberg–Marquardt is performed efficiently. Therefore, the computational complexity of LTRMA-S3 is $O[t * (N_s * N * G + G^3 + N_l * N * N_f)]$.

Similar with LTRMA-MLE, when the number of objects (i.e., N_o) at each ranking list is small, the total learning procedures are efficient for LTRMA-S1, LTRMA-S2, and LTRMA-S3. Nevertheless, when N_o is large, N_s must be quite large (e.g., $\geq 10e6$) because a small sampled ranking set cannot represent the distributions of all candidate rankings. The next section introduces how to learn for ranking lists with a large value of N_o.

5. EXTENSION TO TOP-K SETTINGS

All the above learning algorithms require sampling permutations (i.e., ranking lists in this study) which are used to represent a given ranking distribution in a permutation space. Let N_o be the number of objects in each ranking list. The number of all possible candidate permutations in the permutation space S_{N_o} is $N_o!$. When N_o is large (e.g., 500), a small number (e.g., 1000) of sampled permutations is insufficient to represent a given ranking distribution, which is called insufficient sampling problem. Some recent studies [Lu and Boutilier 2014] proposed new sampling methods to deal with the case when N_o is large. Although these effective sampling algorithms are available, this study solves the insufficient sampling problem in a more effective way.

In most real learning to rank scenarios, users mainly care about the top-ranked objects when the ranking lists are long (i.e., N_o is large). Furthermore, in labeling, it is relatively easy to collect top-k ranking labels from annotators when N_o is large. Therefore, researchers focused on top-k learning to rank in which only top-k labels instead of the full-ordering labels are used for learning in recent years. Both empirical studies on benchmark datasets and theoretical analysis revealed that learning on top-k ground truth is surely sufficient for ranking [Lan et al. 2013]. Figure 8 shows the experimental comparisons between full and top-10 learning to rank conducted by Lan et al. [2013]. The normalized discounted cumulative gain (NDCG) values achieved by top-10 labels for Ranking SVM, RankNet, and ListMLE are higher than those of full-ordering labels. Therefore, this subsection investigates the learning algorithm if the

top-k labels instead of the full-ordering labels are used when N_o is large. Under the top-k setting, the sampling size N_s depends on k instead of N_o. As k is usually small (e.g., 20), a small number of sampled permutations can effectively represent the whole ranking distributions in a large permutation space.

In the top-k setting, the training set is denoted as $D = \{(x^{(i)}, y^{(i)}_{k,1}, \ldots, y^{(i)}_{k,j}, \ldots, y^{(i)}_{k,G})\}_{i=1}^{N}$, where $y^{(i)}_{k,j}$ stands for a the top-k ranking label provided by the jth annotator. Theoretically, $y^{(i)}_{k,j}$ is actually a set of ranking lists such that the top-k items in these ranking lists are the same and the remaining $|x^{(i)}| - k$ items form different permutations. Likewise, we aim to learn a ranking function parameterized by $\mathbf{w}$ (in Equation (5)) and to infer the expertise degrees $\Theta = \{\theta_1, \ldots, \theta_G\}$.

A new generative process for top-k labels can be constructed similar to those depicted in Figure 4. When the maximum likelihood estimation approach is used, both the P–L and Mallow models for top-k labels should be defined. The P–L model for top-k labels is defined as follows:

$$P\left(y^{(i)}_{k,j} | x^{(i)}, \mathbf{w}\right) = \prod_{\iota=1}^{k} \frac{\exp\left(\mathbf{w}^T x^{(i,y^{-1}(\iota))}\right)}{\sum_{l=\iota}^{k} \exp\left(\mathbf{w}^T x^{(i,y^{-1}(\iota))}\right)}$$

For the mallow model, the Kendall–Tau distance D for two top-k ranking lists π and σ is defined by following the definition in Klementiev et al. [2008b]. To make our work self-contained, this distance is introduced as follows.

Definition 1 (*Top-k Kendall–Tau distance*). Let F_π and F_σ be the be the elements of π and σ respectively. $Z = F_\pi \bigcap F_\sigma$ with $|Z| = z$. $P = F_\pi \backslash F_\sigma$, and $S = F_\sigma \backslash F_\pi$ (note that $|P| = |S| = k - z = r$). Define the augmented ranking $\widetilde{\pi}$ as π augmented with the elements of S assigned the same index $k+1$. Clearly, $\widetilde{\pi}^{-1}(k+1)$ is the set of elements at position $n+1$ ($\widetilde{\sigma}$ is defined similarly). Let $I(\varsigma) = 1$ if $\varsigma > 0$, and 0 otherwise. Then, $d(\pi, \sigma)$ is the minimum number of the adjacent transpositions needed to turn $\widetilde{\pi}$ to $\widetilde{\sigma}$ as follows:

$$d(\pi, \sigma) = \sum_{i=1, \widetilde{\pi}^{-1}(i) \in Z}^{n} V_i(\widetilde{\pi}, \widetilde{\sigma}) + \sum_{i=1, \widetilde{\pi}^{-1}(i) \notin Z}^{n} U_i(\widetilde{\pi}, \widetilde{\sigma}) + \frac{r(r+1)}{2}, \tag{52}$$

where

$$V_i(\widetilde{\pi}, \widetilde{\sigma}) = \sum_{l=i, \widetilde{\pi}^{-1}(i) \in Z}^{k} I(\widetilde{\sigma}(\widetilde{\pi}^{-1}(i)) - \widetilde{\sigma}(\widetilde{\pi}^{-1}(l))) + \sum_{l \in \widetilde{\pi}^{-1}(k+1)} I(\widetilde{\sigma}(\widetilde{\pi}^{-1}(i)) - \widetilde{\sigma}(l)), \tag{53}$$

and

$$U_i(\widetilde{\pi}, \widetilde{\sigma}) = \sum_{l=i, \widetilde{\pi}^{-1}(l) \in Z}^{k} 1. \tag{54}$$

With the top-k Kendall–Tau distance defined in Equation (50), the proposed algorithms LTRMA-MLE, LTRMA-S1, LTRMA-S2, and LTRMA-S3 can be easily extended into the top-k setting. In each iteration of the EM algorithm in the top-k LTRMA-MLE, LTRMA-S1, LTRMA-S2, and LTRMA-S3 algorithms, the definition of E_θ becomes

$$E_\theta(d) = \frac{ke^\theta}{1 - e^\theta} - \sum_{j=r+1}^{k} \frac{je^{j\theta}}{1 - e^{j\theta}} + \frac{r(r+1)}{2} - r(z+1)\frac{e^{\theta(z+1)}}{1 - e^{\theta(z+1)}}. \tag{55}$$

Because the above definition is also decreasing monotonically, the computational complexities for top-k LTRMA-MLE, LTRMA-S1, LTRMA-S2, and LTRMA-S3 are

nearly the same as those of LTRMA-MLE, LTRMA-S1, LTRMA-S2, and LTRMA-S3. Nevertheless, because the number of k can be much smaller than N_o, a small number of sampled permutations (i.e., N_s) are sufficient. Consequently, the total learning procedure can be performed more efficiently.

6. EXPERIMENTS

This section evaluates the proposed learning approach for both full-ordering and top-k learning to rank from crowdsourcing labels. In the full-ordering setting, three datasets (including a synthetic set, a real-world set which is popular in learning to rank literature, and a web page set for VisC measurement) are used. In each of these three datasets, the number of objects in each instance is not large. In the top-k setting, two benchmark datasets, which have been widely used in previous learning to rank and rank aggregation literature, are used. The numbers of objects in these two datasets are large ($avg. > 500$). The details of each data set are introduced when the set is first used in the experiments.

Section 6.1 evaluates the proposed learning algorithms in the full-ordering setting. The intuitive algorithm LTRMA-D and the ListMLE algorithm with ground-truth labels are used as the competing algorithms. In addition, the learning algorithm for regression under multi-annotators (called LRegMA for brevity) proposed by Raykar et al. [2010] is also used as the competing algorithm. Section 6.2 evaluates the proposed learning algorithms in the top-k setting. The algorithms top-k LTRMA-D, ListMLE, and LRegMA are used as the competing algorithms. In addition, top-k LTRMA-MLE is also used as the baseline algorithm without considering side information.

On each experimental data set, the competing algorithms are implemented on training and validation sets to search for optimal parameters. The learned model is then adopted to rank the test data. Finally, the NDCG [Liu et al. 2007] is calculated. When this metrics has a higher value, a better result is achieved. In each experiment, for Algorithm 1, N_s is set to 500. For Algorithm 2, τ_1 and τ_2 are set to $0.01 * G$ and $0.01 * N_f$, respectively, where G is the number of annotators and N_f is the feature dimension; each entry of $\mathbf{w}(0)$ is set to $1/N_f$; $\Theta(0)$ is randomly initialized; $MaxT$ is set to 200. The parameter η in both LTRMA-S2 and LTRMA-S3 is set to 20. The parameter setting of LRegMA conforms to the setting in the experiments conducted by Raykar et al. [2010]. The implementation of ListMLE is directly borrowed from the Matlab codes compiled by Xia et al. [2008].

6.1. Results of the Proposed Learning Algorithms for Full-Ordering Learning

6.1.1. Results on Synthetic Data. This study constructs synthetic data by following the similar rule applied by Xia et al. [2008]. First, a two-dimensional point (x_1, x_2) is randomly sampled according to the uniform distribution on a square area $[0, 1] \times [0, 1]$. Then a score is assigned to the point (x_1, x_2) using the following rule: $y = x_1 + 10 * x_2 + e$, where y is the score of the point and e is a random variable normally distributed with zero mean and a standard deviation of 0.005. In total, 15 points associated with their scores (ys) are generated in this manner. Permutation on these scores forms the ranking of the points. The process is repeated to induce 100 training, 100 validation, and 500 testing instances as well as their ground-truth labels. In calculating NDCG, the relevance score of the ith ($i = 1, \ldots, 15$) ranked item is $16 - i$.

The crowdsourcing labels are compiled using the following method. A sum of G ($\in [5, 15]$) annotators is assumed to exist. The expertise degree of the first annotator is set to -2.5, and the remaining values are randomly set in the range of -0.5 to -2.5 (A lower value indicates a higher expertise degree.). Assuming that N training subsets are available and each training instance contains N_o points for ranking. The annotations of G annotators are simulated as follows: for the jth annotator, a number

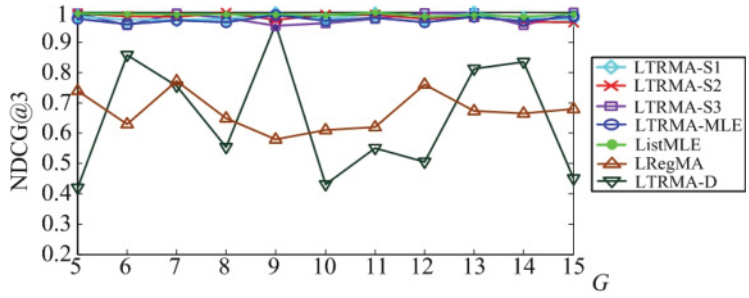

Fig. 9. Performance (NDCG@3) comparison among the competing algorithms on synthetic data.

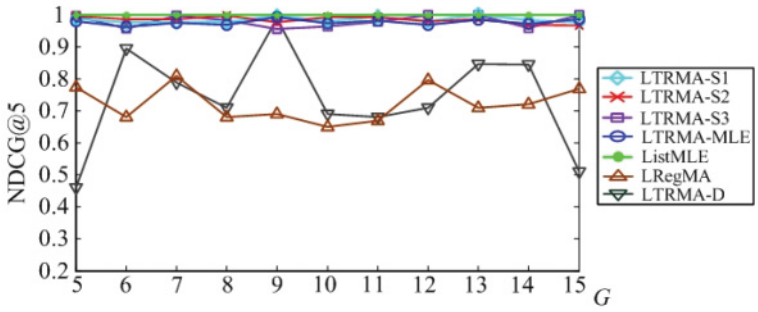

Fig. 10. Performance (NDCG@5) comparison among the competing algorithms on synthetic data.

of N discrete integer values $\{v_{j1}, \ldots, v_{jN}\}$ are first sampled from $[1, N_o(N_o-1)/2]$ with the probability that

$$p(v) = \frac{1}{Z(\theta_j)} \exp(\theta_j \cdot v), \quad v = 1, 2, \ldots, N_o(N_o-1)/2. \tag{56}$$

For the ith training instance $x^{(i)}$, the ranking label $(y_j^{(i)})$ by the jth annotator is subsequently simulated by exchanging two random elements of the ground-truth label $(y^{(i)})$ of $x^{(i)}$ until the Kendall–Tau distance between $y_j^{(i)}$ and $y^{(i)}$ is v_{ji}. With the above method, each training instance obtains G ranking labels, which can be seen as being obtained from G annotators respectively.

The above generation process for both the instances and their associated labels is repeated for 10 times. Ten training, validation, and testing corpora are then obtained. The competing algorithms are performed on the training (including validation) corpus and the NDCG results for each algorithm are recorded. For the LTRMAL-S1 algorithm, the first annotator is assumed to provide the ground-truth labels and 25% proportion of the training instances' true ranking labels are given. When the LRegMA algorithm [Raykar et al. 2010] is used, the score of $x_j^{(i)}$ is set to $15 - i$.

Figures 9 and 10 show the average performances of the competing algorithms in terms of the NDCG@3 and NDCG@5 values on the predicted ranking labels. By conducting the t-test, the proposed four algorithms, LTRMA-MLE, LTRMLE-S1, LTRMLE-S2, and LTRMLE-S3, significantly outperform the competing algorithms LTRMA-D and LRegMA ($p < 0.01$) on almost all values of G. The results in Figures 9 and 10 also reveal that LTRMA-D is not robust. For example, when 9 annotators are simulated, the performance of LTRMA-D is satisfactory. Yet, it poorly performs when only 10 or 12

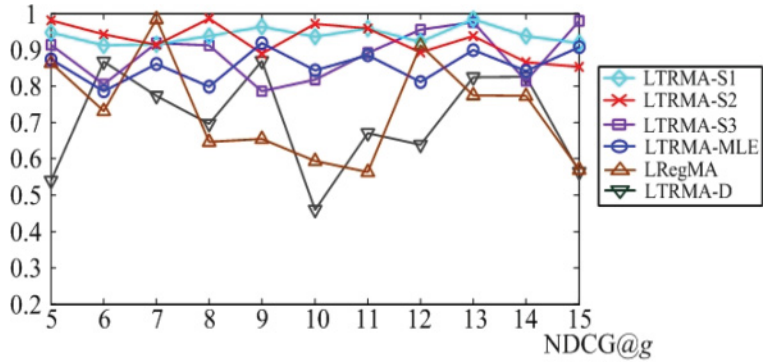

Fig. 11. Performance (NDCG@5) comparison for the estimated expertise degrees among the competing algorithms on synthetic data.

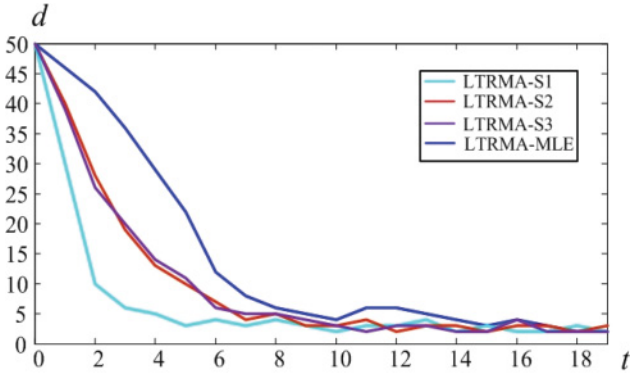

Fig. 12. The convergence of the competing algorithms in terms of the (average) Kendall–Tau distance (d) between the selected best ranking samples to the ground-true ranking labels at each EM iteration.

annotators are simulated. Figure 11 shows the average NDCG performances in terms of the predicted ranking expertise degrees. Figure 11 specifies that the proposed four algorithms, LTRMA-MLE, LTRMLE-S1, LTRMLE-S2, and LTRMLE-S3, also significantly outperform LTRMA-D and LRegMA in terms of the estimated expertise degrees of simulated annotators ($p < 0.01$) based on the t-test. For our proposed four algorithms, LTRMA-MLE is obviously inferior to the other three ones (i.e., LTRMLE-S1, LTRMLE-S2, and LTRMLE-S3) on most numbers of G ($p < 0.05$). LTRMA-S1 and LTRMA-S2 achieve comparable performances and both outperform LTRMA-S3 on most numbers of G ($p < 0.05$).

To compare the convergence rates of the involved algorithms, at the end of each EM iteration, we select the best ranking samples for each algorithm (according to the probability $P(y|\cdot)$ defined in each EM iteration for each algorithm), and compute the average Kendall–Tau distance (d) between the selected best ranking samples and the ground-true ranking labels. Figure 12 shows the convergence for the four proposed algorithms according to the Kendall–Tau distances when sampling is used to estimate the RHS values based on the sampled ranking samples. The convergence of LTRMA-S1 is much faster than those of the other three algorithms LTRMA-S2, LTRMA-S3 and LTRMA-MLE. The convergence rates of LTRMA-S2 and LTRMA-S3 are close to each other and they both are faster than that of LTRMA-MLE. The results

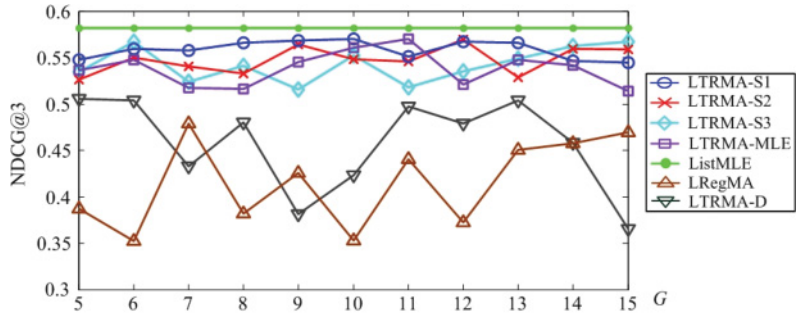

Fig. 13. Performance (NDCG@3) comparison among the competing algorithms on OHSUMED data.

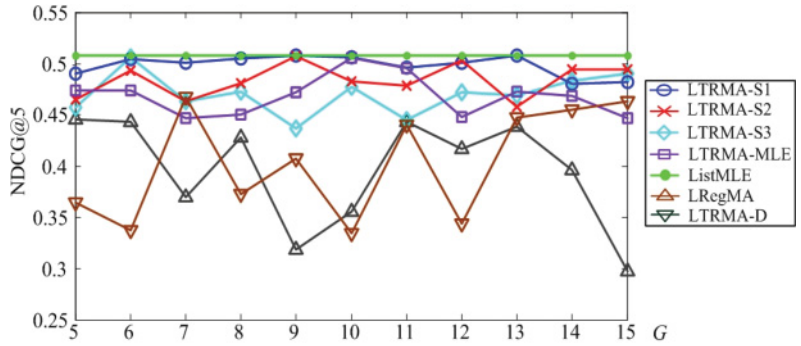

Fig. 14. Performance (NDCG@5) comparison among the competing algorithms on OHSUMED data.

indicate that the side information can reduce the computational complexity of the optimization procedure in learning.

6.1.2. Results on OHSUMED Data. OHSUMED data collection is a benchmark set for learning to rank and is provided in LETOR [Liu et al. 2007]. There are 106 queries on this data corpus. Each query is associated with a set of query–document pairs upon which relevant judgments are made. The degree of relevance is divided into three categories, namely, definitely relevant (score of 3), possibly relevant (score of 2), and not relevant (score of 1). There are 45-dimensional features for each query–document pair. The data separated by LETOR are used to conduct five-fold cross validation.

This study constructs ground-truth labels similar to the manner completed by Xia et al. [2008] and Cao et al. [2007], that is, one perfect permutation among all possible perfect permutations is randomly selected for each query based on ground truth. The labels by annotators are simulated by using the same rule applied on synthetic data. In this experiment, the number of G annotators ranges from 5 to 15. The settings for the competing algorithms are the similar to those for the synthetic data.

The values of NDCG@3 and NDCG@5 for the estimated ranking labels on the test data are displayed in Figures 13 and 14, respectively. Similar to the observations drawn from Figures 9 and 10, the performances of the proposed four algorithms, LTRMA-MLE, LTRMLE-S1, LTRMLE-S2, and LTRMLE-S3, are significantly better than those of LTRMA-D and LRegMA on most G values ($p < 0.01$) by conducting the t-test. The results achieved by LTRMLE-S1 are close to those of ListMLE in both Figures 13 and 14. LTRMLE-S2 and LTRMLE-S3 achieve similar results and slightly outperform LTRMA-MLE on most values of G in both Figures 13 and 14. The performances of LTRMA-D and LRegMA are close to each other. The NDCG values vary dramatically

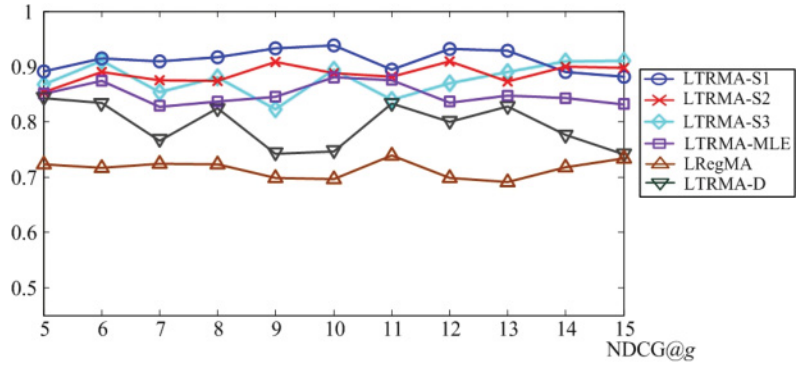

Fig. 15. Performance (NDCG@3) comparison among the competing algorithms in terms of the ranking of the estimated expertise degrees of the annotators on OHSUMED data.

with the increasing of G. Figure 15 shows the performance comparison in terms of the ranking of the estimated expertise degrees of the annotators. The performance comparison results are similar to those in Figures 13 and 14. LTRMA-S1 achieves the best performances ($p < 0.05$). LTRMA-S2 and LTRMA-S3 are slightly better than LTRMA-MLE on most G values.

6.1.3. Results on Web Visual Clutter (VisC) Ranking. As presented in Section 1, web VisC ranking orders web pages according to their VisC features based on a VisC ranking function. The input of the VisC ranking function is the VisC features extracted from a web page. Because web VisC ranking is used in web search, the computational load of the whole VisC feature extraction and ranking approach should be as low as possible. Current VisC measuring algorithms are designed for images, so the features are extracted directly from images, leading that the computational loads of these algorithms are high. Therefore, these algorithms are unsuitable for web search applications even if they can achieve highly accurate measurement (ranking) results. Therefore, this experiment investigates the performances of several features, which can be extracted efficiently, based on the "teaching" of existing image VisC measurement algorithms. If the performance of these efficient features is close to that of image-based algorithms, these efficient features have high potential values for real web applications.

The experimental web pages used in studies on web appearance evaluation are usually home pages. In this experiment, 2000 homepages are collected mainly from the websites of companies and universities as well as some personal sites. For each page, eight simple yet efficient features, including number of texts, number of linked texts, number of fonts, average font size, number of tables, number of background colors, number of images, and aspect ratio, are extracted from the source code and the screenshots of the pages. The average feature extraction time for the eight features of the collected pages is less than 0.005s[9] using a computer with Intel i7-2600 3.4GHz CPU and 4GB memory. All collected pages are divided into 200 subsets, and each subset consists of 10 pages. Three state-of-the-art image VisC measuring algorithms, namely, subband entropy (SE), feature congestion (FC), and segment measuring (SM), are taken as three annotators. The features used in these three image-based algorithms are extracted directly from the screenshots of the collected pages. The average feature extraction time for these three image VisC measuring algorithms is 1.3, 1.9, and 1.5s,[10] respectively.

[9]Because search engines usually save the screenshots of the indexed web pages, the time for the transformation of a web page into a screenshot is not included.
[10]The time for the transformation of a web page into a screenshot is also not included.

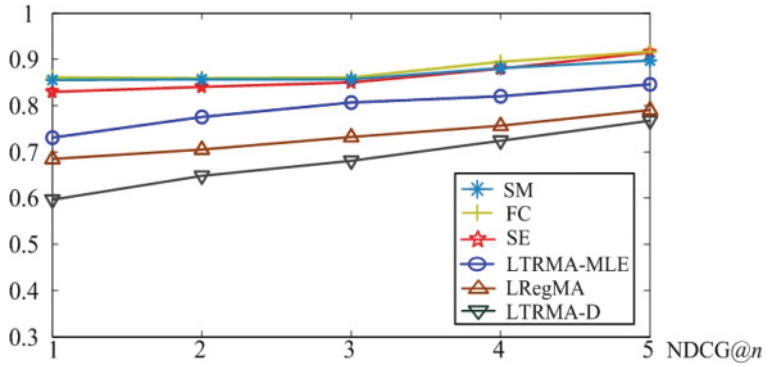

Fig. 16. Results on web visual clutter ranking.

The average feature extraction time of the image VisC measuring algorithm is more than 200 times than that of the extraction of the eight features from the source code. Further details about these methods can be viewed in the research of Bravi and Farid [2008] and Rosenholtz et al. [2007]. Consequently, each subset corresponds to three ranking lists for each training instance (each of the 200 subsets) as given by the three algorithms.

The 200 subsets are randomly divided into two heads: one for training and the other for testing. For each training instance (i.e., a subset consists of 10 web pages), the three image-based VisC measuring algorithms take turns to score each of the 10 pages in the instance. Each image VisC measuring algorithm can yield a ranking list based on its predicted scores on the 10 pages in the instance. Each training instance obtains three ranking lists. That is, each training instance corresponds to three ranking labels which can be seen as being from three annotators. A ranking function can thus be learnt by using the competing algorithms (except the ListMLE because the ground-truth ranking labels in the training set are unavailable) based on the training instances. The test set is used for assessing the performances of the competing algorithms. To generate the ground-truth ranking labels for test data, the pages in the test sets are evaluated by selecting and scoring 10 random subsets by seven volunteers aged 20–30 years who were recruited from the laboratory. The score ranges from 0 to 5; a higher score corresponds to a lower VisC. The training and testing process is repeated 10 times, and the average values of NDCG are reported.

The performances of LTRMA-MLE, LTRMA-D, and LRegMA are indicated in Figure 16, which also presents the NDCG values of the three image-based VisC measuring algorithms SM, FC, and SE. The three proposed algorithms LTRMA-S1, LTRMA-S2, and LTRMA-S3 are not included in Figure 16 due to the reason that the training data compiled above do not contain the corresponding side information. ListMLE is not run because the ground-truth labels are not given. The observations in Figure 16 suggest that LTRMA-MLE significantly outperforms LTRMA-D and LRegMA ($p < 0.01$) based on the t-test. In particular, the performance of LTRMA-MLE is close to those of the three image-based VisC measuring algorithms on NDCG@3, NDCG@4, and NDCG@5. The average difference on the NDCG values is less than 0.1.

Existing crowdsourcing studies [Kajino et al. 2012; Hu et al. 2014] suggest that the injection of partial ground-truth labels can improve the learning performance. To further improve the performance of the efficient features, partial ground-truth labels are injected. The learning context with partial ground-truth labels is just the learning context of LTRMA-S1. Fifty test instances with ground-truth labels are moved from the above test set to the training set. The new performances are shown in Figure 17.

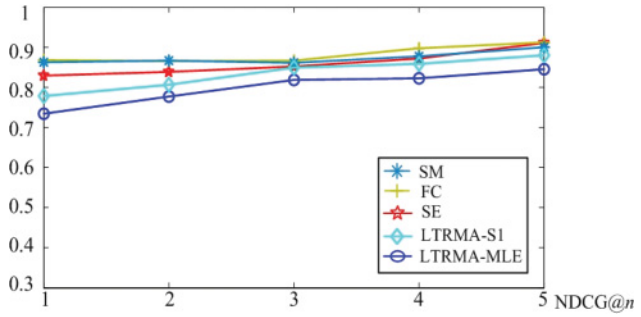

Fig. 17. Results on web visual clutter ranking when some ground-truth labels are injected into the training set.

By conducting the t-test, LTRMA-S1 is obviously better than LTRMA-MLE on all the NDCG values in Figure 17 ($p < 0.01$). The injection of ground-truth labels does improve the learning performance. Considering that the computational complexities for the features used in LTRMA-MLE are much lower than those for the features used in the image-based algorithms while the performance of the former is close to the latter, the former algorithm is meaningful for the real VisC measuring applications.

6.2. Results of the Proposed Algorithms in the Top-K Setting

This subsection evaluates the proposed algorithms in the top-k setting when the ranking lists in training are long. Two benchmark datasets are used in the following experiments with slightly different experimental settings[11] based on the compiling of the MQ2007 and MQ2008 datasets in LETOR 4.0[12] [Liu 2011].

6.2.1. Experiments on MQ2007 Data. The MQ2007 data set is provided by the LETOR 4.0 package. It is compiled based on Gov2 Web page collection and two query sets from Million Query track of TREC 2007. There are 1694 queries in MQ2007 with ranked documents. There are 45-dimensional features for each query–document pair. The histograms of the numbers of the ranked objects (i.e., ranked documents) for each training instance (i.e., query) are shown in Figure 18. Most training instances contain more than 500 objects. The five-fold cross validation strategy is adopted and the five-fold partition setting in LETOR is followed. In each fold, there are three subsets for learning: training, validation and testing. As the listwise ranking labels are available, it is unnecessary to generate ranking labels as the OHSUMD set does.

To run the proposed methods, the data used in the experiments are compiled as follows. First, the queries with more than 500 ranked documents are kept. Second, for each query in the training subset, the objects (query–document pairs) with an even number are extracted to form a new ranking list. We then obtain a new training subset. Third, we construct eight ranking functions with eight different types of features listed in Table I by using the ListMLE algorithm. These eight ranking functions can be seen as eight annotators. These eight ranking functions are evaluated on the ground-truth labels for the objects with an odd numbers in the training set. The performance ordering

[11]The rank aggregation data set compiled by LETOR 4.0 package [Liu 2011] consists of the query–document instances and multiple ranking lists. However, for all the training instances, each input ranking label contains partial rankings and the contained objects in many input ranking labels are limited and different. This work assumes that each ranking labels contain equal objects. As a consequence, the LETOR rank aggregation data set is not used. The labels in the crowdsourcing task in TREC 2011 are the ordinal ranking for the objects, whereas this work focuses on listwise learning to rank.
[12]http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4dataset.aspx.

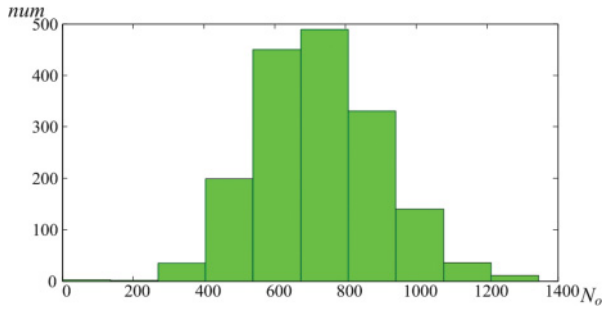

Fig. 18. The history of the number of objects contained in the training instances in the MQ2007 data set.

Table I. The Eight Subsets of Features Used for the Construction of Eight Ranking Functions

Ranking function-1	TFIDF features in abstract
Ranking function-2	TFIDF features in "title + abstract"
Ranking function-3	TFIDF + BM25 features in abstract
Ranking function-4	TFIDF + LMIR features in abstract
Ranking function-5	TFIDF + BM25 features in "title + abstract"
Ranking function-6	TFIDF + LMIR features in "title + abstract"
Ranking function-7	TFIDF + BM25 + LMIR features in abstract
Ranking function-8	TFIDF + BM25 + LMIR features in both abstract and "title + abstract"

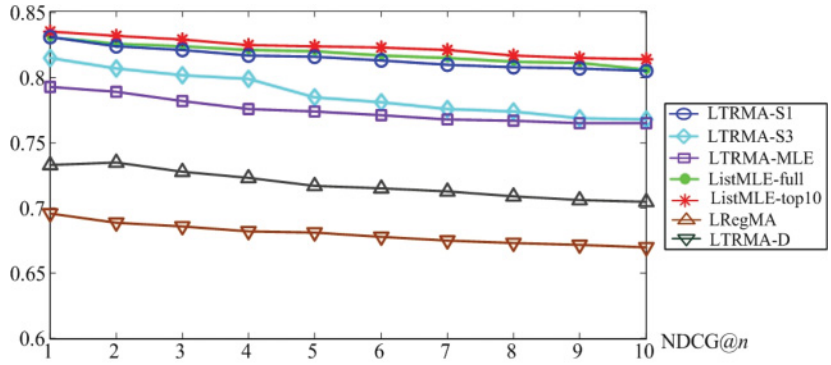

Fig. 19. NDCG on the MQ2007 set in the presence of Types I and III side information.

for the eight functions is thus obtained. Ranking function-8 achieves the highest NDCG results, which is used as the side information in the experiments. They are used to rank the left objects in the training subset and each subset obtains eight ranking labels. In the performing of the proposed algorithms, all the 45-dimensional features and the rank labels by the eight ranking functions are used. The LTRMA-S2 algorithm is not used in the experiments because the above compiling approach does not produce the range of any annotator's expertise degree.

Figure 19 shows the results of the competing algorithms in terms of the NDCG values on the predicted ranking lists on the test data. The performance of LTRMA-S1 is quite close to those of ListMLE-full and ListMLE-top10. These three algorithms are significantly better than the rest competing algorithms ($p < 0.01$) according to the t-test. LTRMA-S3 is better than LTRMA-MLE especially on NDCG@1–4. The two algorithms LTRMA-D and LRegMA achieve the worst performances. Figure 20 shows the competing results in terms of the NDCG values on the ranking of the predicted

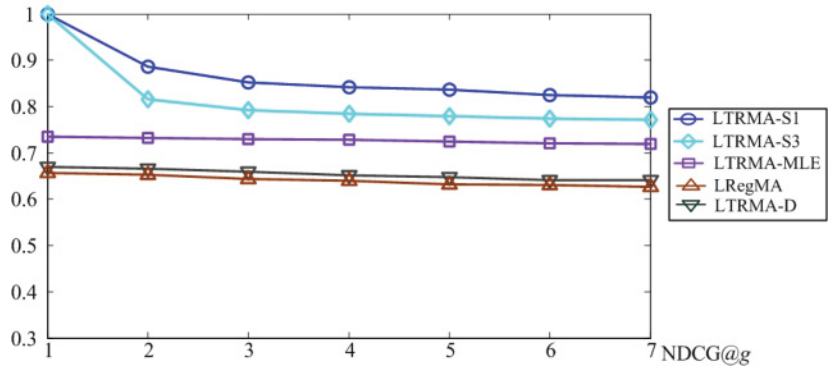

Fig. 20. NDCG on the ranking of the predicted expertise degrees on the MQ2007 set in the presence of Types I and III side information.

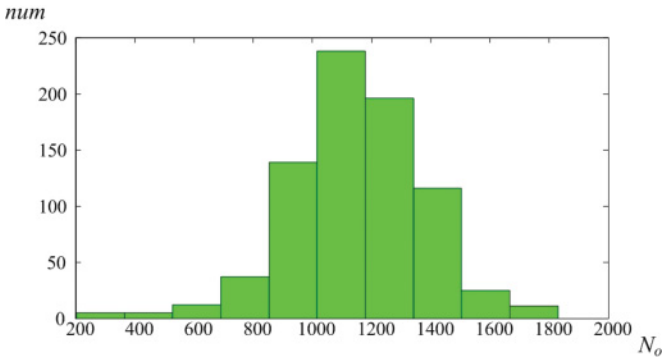

Fig. 21. The history of the number of objects contained in the training instances in the MQ2008 data set.

expertise degrees. Naturally, both the NDCG@1 values of LTRMA-S1 and LTRMA-S3 are equal to 1. The overall performance of LTRMA-S1 is better than that of LTRMA-S3 ($p < 0.01$). The NDCG values achieved by LTRMA-S1 and LTRMA-S3 on all the G numbers are significantly higher than those achieved by the rest competing algorithms ($p < 0.01$). In addition, for all the proposed algorithms, the maximum iterations of the EM optimization are smaller than 20. Therefore, the computational load for all the proposed algorithms is low. Each experimental run on each fold of training is finished within 20min.

6.2.2. Experiments on MQ2008 Data. The MQ2008 data set is also compiled based on Gov2 web page collection and two query sets from Million Query track of TREC 2007. There are about 800 queries in MQ2008 with ranked documents. There are 45-dimensional features for each query–document pair. The histograms of the numbers of the ranked objects (i.e., ranked documents) for each training instance (i.e., query) are shown in Figure 21. Most training instances contain more than 500 objects. The five-fold cross validation strategy is adopted and the five-fold partition setting in LETOR is followed. The training, validation and test datasets used in the experiments are compiled using the similar way to that used for the MQ2007 data. The LTRMA-S2 algorithm is still not used in the experiments because the above compiling approach does not produce the range of any annotator's expertise degree.

Figure 22 shows the results of the competing algorithms in terms of the NDCG values on the predicted ranking lists. By conducting the t-test, the performance of

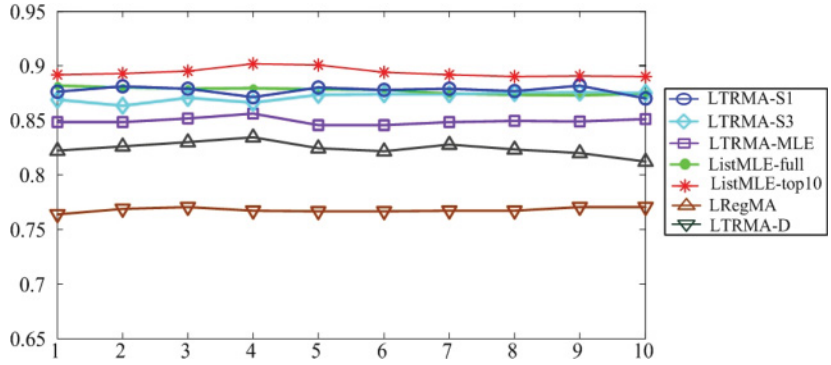

Fig. 22. NDCG on the MQ2008 data in the presence of Types I and III side information.

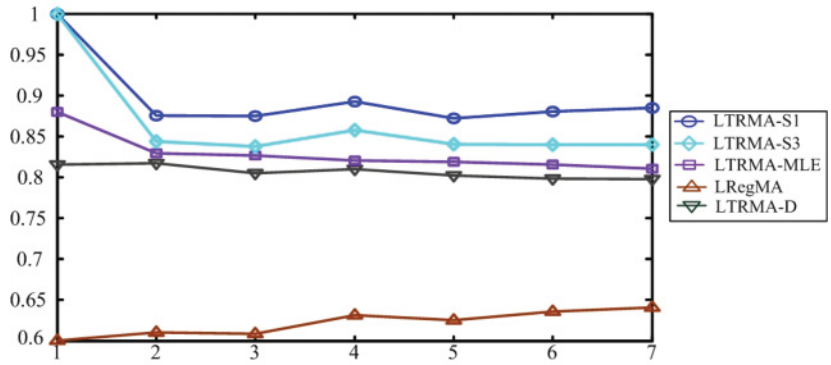

Fig. 23. NDCG on the ranking of the predicted expertise degrees on the MQ2008 set in the presence of Types I and III side information.

ListMLE-top10 is better than those of all the other involved algorithms ($p < 0.01$), indicating that learning to rank with only top labeled ranking lists is effective. The three algorithms LTRMA-S1, LTRMA-S3, and ListMLE-full obtain similar NDCG values. LTRMA-MLE is inferior to these three algorithms, whereas it outperforms LTRMA-D and LRegMA ($p < 0.01$). Figure 23 shows the performances of the predicted expertise degrees achieved by the competing algorithms. The comparisons among the involved algorithms are consistent with those from Figure 22. Side information does improve the learning performances. For all the proposed algorithms, the computational load is acceptable as the learning procedure is converged in no more than 15 iterations. Each experimental run on each fold of training is finished within thirty minutes.

6.3. Discussion

The experimental results in Section 6.1 reveal that the proposed algorithms (LTRMA-MLE, LTRMA-S1, LTRMA-S2, and LTRMA-S3) are significantly better than LTRMA-D. The probable reason for this instance is that the proposed algorithms unify both ground truth estimation and model learning in a probabilistic formulation, which models their relationships in an improved manner. The proposed algorithms are more robust than LTRMA-D in terms of the varying number of annotators. This circumstance is supported by the reason that in LTRMA-D, model learning is significantly sensitive to ground truth estimation in the first step, whereas in the proposed algorithms, model learning can respond to the ground truth estimation in the next iteration. The results

of web VisC data indicate that considering various simple and efficient features can induce a performance close to those of state-of-the-art algorithms, which are based on image content analysis. Hence, if an application that involves web VisC ranking is time sensitive, considering only some efficient features can probably induce beneficial implications. In all the experiments, the proposed algorithms (i.e., LTRMA-S1, LTRMA-S2, and LTRMA-S3) which utilize the side information are superior to LTRMA-MLE which does not. In addition, the algorithm LRegMA, which is directly borrowed from existing crowdsourcing machine learning studies and simply takes ranking positions as scores, obtains the worst results. The results on the particular case of Type I side information suggest that partial true labels improve the learning performance. The comparative results imply that Types II and III side information, although they are more passive than Type I side information, can also improve learning performances, particularly the estimated expertise degrees of annotators.

The experimental results in Section 6.2 demonstrate that, when each training instance contains a large number of objects (query–document pairs in the experiments), the proposed algorithms (LTRMA-S1, LTRMA-S2, and LTRMA-S3) can also achieve effective results based on only top-10 ranked objects in the ranking labels compared with LTRMA-D and LRegMA. Meanwhile, the computational consumption is low as the computational complexity depends largely on k instead of the length of the full-ordering lists.

7. CONCLUSIONS

This study investigates learning to rank under multiple annotators, which provide labels that are not completely accurate. A new learning approach (LTRMA-MLE), which iteratively estimates ground truth and uses conventional algorithms (e.g., ListMLE) in training a ranking function based on estimated ground truth, is proposed. LTRMA-MLE integrates both ground truth estimation and ranking function learning with a maximum likelihood estimation framework. Experiments suggest that LTRMA-MLE outperforms the direct approach (LTRMA-D) and is very close to the performance of ListMLE, which employs ground-truth labels. The experiment on web VisC data indicates that, based on LTRMA-MLE, a VisC ranking function whose performance is close to that of state-of-the-art image VisC measuring algorithms can be constructed by considering only simple features which can be extracted quite efficiently. If some true labels are injected into the training set, the performance can be further improved.

Side information (e.g., credit records, professional grades, and history of annotation accuracy) about annotators is available in many crowdsourcing labeling tasks. Accordingly, three basic types of side information about the expertise degrees of annotators are summarized in the paper. These types of side information are incorporated into LTRMA-MLE by subsequently proposing three new algorithms, namely, LTRMA-S1, LTRMA-S2, and LTRMA-S3. Complex side information can be depicted by the combination of the three basic ones. Experimental results signify that the proposed learning algorithms can obtain better results than LTRMA-MLE without considering side information. Considering that the sampling for rankings in a large permutation space is difficult when each training instance contains a large number of objects, the top-k learning to rank from crowds is also investigated. Experiments suggests that the learning with only top-k rankings is efficient and the results are comparable to those of learning with ground-truth full-ranking labels.

ACKNOWLEDGMENTS

We thank Mr Jun Gao for his very useful comments and suggestions on the English writings of the paper.

REFERENCES

C. C. Aggarwal, Y. Zhao, and P. S. Yu. 2012. On text clustering with side information. In *Proceedings of IEEE 28th International Conference on Data Engineering (ICDE)*. 894–904.

M. J. Bravi and H. Farid. 2008. A scale invariant measure of clutter. *J. Vis.* 8, 1 (2008), 1–9.

Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. 2007. On text clustering with side information. In *Proceedings of International Conference on Machine Learning (ICML)*. 129–136.

Q. Chen, Z. Song, Y. Hua, Z. Huang, and S. Yan. 2012. Hierarchical matching with side information for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3426–3433.

S. Chen, J. Zhang, G. Chen, and C. Zhang. 2010. What if the irresponsible teachers are dominating? A method of training on samples and clustering on teachers. In *Proceedings of the 25th National Conference on Artifical Intelligence (AAAI'10)*. 419–424.

W. Cheng, K. Dembczyń, and E. Hüllermeier. 2010. Label ranking methods based on the Plackett–Luce model. In *Proceedings of International Conference on Machine Learning (ICML)*. 215–222.

X.-Q. Cheng, P. Du, J. Guo, X. Zhu, and Y. Chen. 2013. Ranking on data manifold with sink points. *IEEE Trans. Know. Data Eng.* 25, 1 (2013), 177–191.

O. Dekel and O. Shamir. 2009. Vox Populi: Collecting high-quality labels from a crowd. In *Proceedings of Annual Conference on Learning Theory (COLT)*. 1–6.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc.: Ser. B* 39, 1 (1977), 1–38.

P. Donmez and J. Garbonell. 2010. A probabilistic framework to learn from multiple annotators with time-varying accuracy. In *Proceedings of SIAM International Conference on Data Mining (SDM)*. 862–871.

X. Geng, T. Qin, T.-Y. Liu, and X.-Q. Cheng. 2012. A noise-tolerant graphical model for ranking. *Inf. Process. Manag.* 48, 2 (2012), 374–383.

Q. He, J. Si, and D. J. Tylavsky. 2000. Prediction of top-oil temperature for transformers using neural networks. *IEEE Trans. Power Deliv.* 15, 4 (2000), 1205–1211.

Q. Hu, Q. He, H. Huang, K. Chiew, and Z. Liu. 2014. Learning from crowds under experts' supervision. In *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. 2000–2011.

H. Kajino, Y. Tsuboi, I. Sato, and H. Kashima. 2012. Learning from crowds and experts. In *Proceedings of the 4th AAAI Human Computation Workshop (HCOMP)*. 107–113.

A. Klementiev, D. Roth, and K. Small. 2008a. Unsupervised rank aggregation with distance-based models. In *Proceedings of International Conference on Machine Learning (ICML)*. 472–479.

A. Klementiev, D. Roth, K. Small, and I. Titov. 2008b. Unsupervised rank aggregation with domain-specific expertise. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*. 1101–1106.

Y. Lan, S. Niu, J. Guo, and X.-Q. Cheng. 2013. Is top-k sufficient for ranking? In *Proceedings of International Conference on Information and Knowledge Management (CIKM)*. 1261–1270.

G. Lebanon and J. Lafferty. 2002. Cranking: Combining rankings using conditional probability models on permutations. In *Proceedings of International Conference on Machine Learning (ICML)*. 363–370.

B. Liu, X. Li, W. S. Lee, and P. S. Yu. 2004. Text classification by labeling words. In *Proceedings of the 19th National Conference on Artifical Intelligence (AAAI'04)*. 425–430.

T.-Y. Liu. 2011. *Learning to Rank for Information Retrieval*. Springer, Berlin, 285 pages.

T.-Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. 2007. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR Workshop on LTR4IR*.

T. Lu and C. Boutilier. 2011. Learning Mallows models with pairwise preferences. In *Proceedings of International Conference on Machine Learning (ICML)*. 145–152.

T. Lu and C. Boutilier. 2014. Effective sampling and learning for Mallows models with pairwise-preference data. *J. Mach. Learn. Res.* 15 (2014), 3783–3829.

R. D. Luce. 1959. *Individual Choice Behavior: A Theoretical Analysis*. Wiley, New York, NY.

C. L. Mallow. 1957. Non-null ranking models. *Biometrika* 44 (1957), 114–130.

T. Matsui, Y. Baba, T. Kamishima, and H. Kashima. 2014. Crowdordering. In *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. 336–347.

G. McLachlan and T. Krishnan. 1996. *The EM Algorithm and Extensions*. John Wiley & Sons, New York, NY.

J. Nocedal and S. J. Wright. 2006. *Numerical Optimization*. Springer, Berlin.

R. Plackett. 1975. The analysis of permutations. *Appl. Stat.* 24 (1975), 193–202.

T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, and H. Li. 2008. Global ranking using continuous conditional random fields. In *Proceedings of Neural Information Processing Systems (NIPS)*. 1281–1288.

T. Quin, X. Geng, and T.-Y. Liu. 2010. A new probabilistic model for rank aggregation. In *Proceedings of Neural Information Processing Systems (NIPS)*. 1948–1956.

V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. 2010. Learning from crowds. *J. Mach. Learn. Res.* 11 (2010), 1297–1322.

R. Rosenholtz, Y. Li, and L. Nakano. 2007. Measuring visual clutter. *J. Vis.* 72, 2 (2007), 1–22.

C. Shen and T. Li. 2011. Learning to rank for query-focused multi-document summarization. In *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM)*. 626–634.

K. Small, B. C. Wallace, C. E. Brodley, and T. A. Trikalinos. 2011. The constrained weight space SVM: Learning with ranked features. In *Proceedings of International Conference on Machine Learning (ICML)*. 754–763.

P. Smyth, U. Fayyad, M. Burl, P. Perona, and P. Baldi. 1995. Inferring ground truth from subjective labelling of venus images. In *Proceedings of Neural Information Processing Systems (NIPS)*. 1085–1092.

H. A. Soufiani, W. Chen, D. C. Parkes, and L. Xia. 2013. Generalized method-of-moments for rank aggregation. In *Proceedings of Neural Information Processing Systems (NIPS)*. 2706–2714.

M. N. Volkovs, H. Lorochelle, and R. S. Zemel. 2012. Learning to rank by aggregating expert preferences. In *Proceedings of International Conference on Information and Knowledge Management (CIKM)*. 843–851.

M. N. Volkovs and R. S. Zemel. 2012. A fexible generative model for preference aggregation. In *Proceedings of International World Wide Web Conference (WWW)*. 479–488.

M. N. Volkovs and R. S. Zemel. 2014. New learning methods for supervised and unsupervised reference aggregation. *J. Mach. Learn. Res.* 15 (2014), 1135–1176.

L. Wu, S. C. H. Hoi, R. Jin, J. Zhu, and N. Yu. 2011a. Distance metric learning from uncertain side information for automated photo tagging. *ACM Trans. Intell. Syst. Technol.* 2 (2011), 1–28.

O. Wu, W. Hu, and J. Gao. 2011b. Learning to rank under multiple annotators. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*. 1571–1576.

F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. 2008. Listwise approach to learning to rank - Theory and algorithm. In *Proceedings of International Conference on Machine Learning (ICML)*. 1192–1199.

S. Xie, W. Fan, and P. S. Yu. 2012. An iterative and re-weighting framework for rejection and uncertainty resolution in crowdsourcing. In *Proceedings of SIAM International Conference on Data Mining (SDM)*. 1107–1118.

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. 2003. Distance metric learning with application to clustering with side-information. In *Proceedings of Neural Information Processing Systems (NIPS)*. 505–512.

M. Xu, R. Jin, and Z.-H. Zhou. 2013. Speedup matrix completion with side information: Application to multi-label learning. In *Proceedings of Neural Information Processing Systems (NIPS)*. 2301–2309.

Y. Yan, R. Rosales, G. Fung, S. Ramanathan, and G. D. Jennifer. 2014. Learning from multiple annotators with varying expertise. *Mach. Learn.* 95, 3 (2014), 291–327.

Y. Yan, R. Rosales, G. Fung, M. Schmidt, G. Hermosillo, G. Bogoni, L. Moy, and J. G. Dy. 2010. Modeling annotator expertise: learning when everybody knows a bit of something. In *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTAT)*. 932–939.

Y. Zhao and P. S. Yu. 2013. On graph stream clustering with side information. In *Proceedings of SIAM International Conference on Data Mining (SIAM SDM)*. 139–150.

Received September 2014; revised October 2015; accepted March 2016