Listwise Approach for Rank Aggregation in Crowdsourcing

Shuzi Niu[†*], Yanyan Lan[‡], Jiafeng Guo[‡], Lei Yu[†] and Guoping Long[†], Xueqi Cheng[‡]
[†]Institute of Software, Chinese Academy of Sciences, Beijing, P. R. China
[‡]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, P. R. China
{shuzi,yulei,guoping}@iscas.ac.cn, {lanyanyan,guojiafeng,cxq}@ict.ac.cn

ABSTRACT

Inferring a gold-standard ranking over a set of objects, such as documents or images, is a key task to build test collections for various applications like Web search and recommender systems. Crowdsourcing services provide an efficient and inexpensive way to collect judgments via labeling by sets of annotators. We thus study the problem of finding a consensus ranking from crowdsourced judgments. In contrast to conventional rank aggregation methods which minimize the distance between predicted ranking and input judgments from either pointwise or pairwise perspective, we argue that it is critical to consider the distance in a listwise way to emphasize the position importance in ranking. Therefore, we introduce a new listwise approach in this paper, where ranking measure based objective functions are utilized for optimization. In addition, we also incorporate the annotator quality into our model since the reliability of annotators can vary significantly in crowdsourcing. For optimization, we transform the optimization problem to the Linear Sum Assignment Problem, and then solve it by a very efficient algorithm named CrowdAgg guaranteeing the optimal solution. Experimental results on two benchmark data sets from different crowdsourcing tasks show that our algorithm is much more effective, efficient and robust than traditional methods.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Relevance Feedback

General Terms

Algorithms

*This work was conducted when the first author was a Ph.D student at Institute of Computing Technology, Chinese Academy of Sciences.

WSDM'15, February 2–6, 2015, Shanghai, China.

http://dx.doi.org/10.1145/2684822.2685308 .

Keywords

Crowdsourced Labeling; Rank Aggregation; Evaluation Measures

1. INTRODUCTION

Inferring ranking over a set of objects is a critical task for building gold-standard test collections in many ranking-based real applications, such as information retrieval and recommender systems. Recently, crowdsourcing services have attracted much attention since it provides an inexpensive and efficient means to obtain judgments over the objects. Typically, there are two kinds of judgments widely adopted in crowdsourcing, i.e. ratings and preferences. For example in a query-document relevance labeling task, annotators are asked to present the rating for each document such as binary or graded relevance judgments with absolute labeling strategy [13], while they are asked to present preferences for each document pair with relative labeling strategy [3].

In literature, many rank aggregation methods have been proposed to find a consensus ranking over these judgments. They all fall into the framework of minimizing a distance between predicted ranking and input judgments. According to different distances used for optimization, they can be mainly divided into three categories: pointwise, pairwise and listwise approaches. Pointwise rank aggregation methods like MedianRank [7] utilize Footrule Distance as the objective function, which aims to well predict the rank of each object from annotators' judgments on that object. Pairwise methods such as Bradley-Terry [23], MPM [26] and CrowdBT[4] measure the distances in a pairwise way, and pairs are viewed as independent. We can see that both pointwise and pairwise approach define the distance from a local perspective. Therefore, they all ignore the position importance in ranking, which is nevertheless critical for rank aggregation. Traditional listwise rank aggregation methods such as Cranking [16] and Plackett-Luce [10] are infeasible for most crowdsourced labeling tasks, such as crowdsourced pairwise labeling task.

Therefore we propose a novel listwise approach to tackle this problem. The key idea is that in order to take position importance into account, it is better to define the distance in a listwise way. Inspired by the fact that IR measures such as NDCG and RBP are designed for this purpose, and usually used for evaluation in rank aggregation, we propose to directly utilize these measures for distance definition. However, the characteristics of crowdsourcing poses two challenges to this approach: (1) Annotators usually provide ratings or preferences over a subset of objects in crowdsourcing, which

make the computation of ranking measures impossible since the ranking information cannot be induced from the incomplete input information; (2) Annotator quality should be considered in the objective function since the reliability of annotators can vary significantly in crowdsourcing.

To address these challenges, we propose to map the judgments (ratings or preferences) to input ranking, and incorporate annotator quality in this process. Specifically, the rank of an object is viewed as a random variable and defined in the form of pairwise contests. The distribution of the random variable is then derived iteratively based on the pairwise probability, which can be estimated through pairwise preference relationships conveyed in judgments. Meanwhile, annotator quality is modeled as the probability that one agrees with the (unknown) true pairwise preference relationships, which is estimated in an iterative way based on the intermediate aggregation results. By incorporating annotator quality into the definition of the pairwise probability, we obtain the final form of rank distribution of each object. Based on this listwise mapping, we define the new objective functions for rank aggregation as the expectation of IR measures (i.e. NDCG and RBP) over the rank distribution, called expected measures. Note that the ground-truth label of an object in the expected measure is derived from its rank using a mapping function as in [19].

Due to the property of ranking measures [15], we find that expected ranking measures have a general formulation as a sum of utility functions on ranks in both the induced ranking inputs and the output permutations. Therefore, the goal of the optimization approach is to find a ground-truth permutation which maximize the sum of utility functions. As a sequence, it is natural to transform the rank aggregation problem to the Linear Sum Assignment Problem (LSAP for short), where the profit for each assignment is defined by taking the sum of utility functions over all the induced ranking inputs. As the utility functions are represented as a product of two functions, the optimal solution of the optimization problem can be directly obtained through sorting as shown by Rearrangement Inequality. We refer this optimization algorithm as CrowdAgg.

Finally we conduct extensive experiments on two benchmark data sets from different crowdsourcing tasks, i.e. query-document relevance labeling and music similarity labeling. One is collected based on graded judgments, and the other is based on pairwise preferences. Experimental results show that CrowdAgg is much more effective, efficient and robust than traditional rank aggregation methods in both data sets.

In summary, the main contributions of our proposed approach are listed as follows:

- we introduce expected measures as the new objective functions for rank aggregation to consider position importance, by utilizing rating and preference judgments in a listwise way;
- We incorporate annotator quality into the expected measures to well cope with the reliability of annotators in crowdsourcing;
- We propose an efficient algorithm by formalizing the aggregation problem as LSAP and obtaining an optimal solution with Rearrangement Inequality.

2. BACKGROUNDS

In this section, we first give a problem formalization of rank aggregation in crowdsourcing. We then review some related works of traditional pointwise and pairwise rank aggregation methods. Finally, we give a brief introduction to some major evaluation measures used in rank aggregation.

2.1 Rank Aggregation in CrowdSourcing

Assume that we are given a set of n objects denoted as $D = \{x_1, x_2, \ldots, x_n\}$ and a set of m annotators in a crowdsourcing task. Each annotator i provides a set of labels τ_i over these objects, which are often unreliable and incomplete. Typically, each τ_i can be represented in two forms: ratings or preferences.

(1) With absolute labeling strategy, annotators are asked to present the relevance rating of each object. Usually, the ratings are incomplete, which means that annotators may only rate a subset of all the objects. We denote this subset labeled by annotator i as D_i. In this case, τ_i can be represented as $(\tau_i(x_1), \cdots, \tau_i(x_n))$, where

$$\tau_i(x_j) = \begin{cases} z_j^{(i)} \in \{0, 1, \cdots, C-1\}, & \text{if } x_j \in D_i; \\ Null, & \text{if } x_j \notin D_i. \end{cases}$$

(2) With relative labeling strategy, annotator are asked to present the relative relevance between any two objects. Usually, the preferences are also incomplete, which means that annotators may only provide preferences over a subset of all the possible object pairs. We denote this subset labeled by annotator i as P_i. In this case, τ_i can be represented as $(\tau_i(x_1, x_2), \cdots, \tau_i(x_{n-1}, x_n))$, where $\tau_i(x_s, x_t)$ is defined as follows, and $z_j^{(i)} = 1$ means x_s is more relevant than x_t in terms of annotator i; otherwise $z_j^{(i)} = 0$.

$$\tau_i(x_s, x_t) = \begin{cases} z_j^{(i)} \in \{0, 1\}, & \text{if } (x_s, x_t) \in P_i; \\ Null, & \text{if } (x_s, x_t) \notin P_i. \end{cases}$$

The goal of rank aggregation in crowdsourcing is then to find a consensus ranking π over these n objects which best represents the ranking relations conveyed in multiple sets of labels $\{\tau_1, \ldots, \tau_m\}$ in the form of ratings or preferences, where π is a permutation and $\pi(x_i)$ stands for the position of object x_i in the final consensus ranking π. To this end, most aggregation algorithms try to optimize a distance measure M between the inputs $\tau_1, \cdots, \tau_m$ and the final ranking π, which can be formulated as follows.

$$\max_{\pi \in \Pi} \sum_{i=1}^{m} M(\pi, \tau_i). \tag{1}$$

2.2 Rank Aggregation Methods

Here we briefly review some of the traditional rank aggregation methods in unsupervised scenario. According to different kinds of distance measures used for optimization, they can be mainly divided into three categories: pointwise, pairwise and listwise methods.

2.2.1 Pointwise Rank Aggregation Methods

Pointwise rank aggregation methods utilize the label information per object from all the inputs to define the ranking function or objective function. The distance measure between the consensus ranking and the input ranking for Borda Count [1] and Median Rank [7] is decomposed into position difference per object. Borda Count [1] minimizes

the average Spearman Rank Coefficient and obtains the optimal ranking by the mean position of each object. Median Rank [7] optimizes the average Spearman Footrule Distance between the consensus ranking and each ranking input, and obtains the optimal solution by sorting objects according to their median rank in ascending order.

2.2.2 Pairwise Rank Aggregation Methods

Pairwise rank aggregation methods organize their ranking inputs in a pairwise way whether for ranking functions or optimization objective functions.

Graph based Method. Condorcet Fuse [18] constructs the Condorcet Graph with n items and its arc representing the pairwise comparison results between two items by majority voting, and a Hamiltonian path is obtained from this graph by QuickSort. With all the aggregated pairwise preferences summarized in a tournament, GreedyOrder [5] minimizes the pairwise disagreement cost in this tournament to obtain the consensus ranking.

Pairwise Preference Matrix Based Methods. With all the inputs summarized in a pairwise preference matrix, SVP (Singular Vector Projection) [9] minimized nuclear norm of this pairwise preference matrix by rank-2 factorization.

Pairwise Probabilistic Approach. Probabilistic approaches define the generative probability of pairwise preferences and optimize the likelihood function by a gradient based approach [23, 26, 4]. Bradley-Terry [23] defines the pairwise probability based on the Bradley-Terry model [22]. In crowdsourcing, annotator accuracy should be incorporated. MPM (Multinomial Preference Model) [26] models both the power of rank difference and the deviation from the consensus ranking per annotator in the pairwise generative probability. Crowd-BT [4] extends the Bradley-Terry model by explicitly incorporating the labeling qualities of different annotators. The key preprocess in EloRating [2, 27] is to estimate pairwise probabilities and annotator qualities through EM algorithm, then obtain the relevance score from the remaining preferences with the estimated information through Elo-Rating system.

2.2.3 Listwise Rank Aggregation Methods

Listwise rank aggregation methods take the position importance into consideration and treat ranking inputs in a listwise way whether for ranking functions or optimization objective functions. Plackett-Luce [10] and Cranking [16] define the similarity measure to be the generative probability of each ranking list with Plackett-Luce model [22] and Mallow's model [22] respectively, and optimize this similarity function by a maximum likelihood procedure. Although the annotator quality can be incorporated into the optimization objective function, these listwise methods do not work when these inputs are in the form of pairwise preferences. St.Agg [20], one of state-of-the-art methods in traditional rank aggregation tasks like metasearch, can solve this problem by deriving one item's rank position from its pairwise preferences with other items, but it is not fit for the crowdsourced setting. So these listwise approaches are not used as baselines in our experiments.

2.3 Evaluation Measure

Given a set of objects $D = \{x_1, \cdots, x_n\}$, and the ground-truth labels $Y = (y_1, \cdots, y_n)$, which are usually based on multi-grade ratings. Let π be the consensus ranking list over D by some rank aggregation method. Evaluation measures such as NDCG [12], Precision [6] and RBP [17] defined as follows are often employed to evaluate the performance of this aggregation method.

$$\text{NDCG@}k(\pi, Y) = \frac{\sum_{j=1}^{n} g(y_j) D(r_j^\pi) I(r_j^\pi \le k)}{\text{DCG}_{\max}(n)}, \tag{2}$$

where r_j^π stands for the rank of x_j in the ranking list π, $g(y_j)$ is the gain function with $g(y_j) = 2^{y_j} - 1$, $D(r_j^\pi)$ is the discount function with $D(r_j^\pi) = \frac{1}{\log(1+r_j^\pi)}$, $I(\cdot)$ is an indicator function with $I(A) = 1$ if A is true and $I(A) = 0$ otherwise.

$$\text{Precision@}k(\pi, Y) = \sum_{j=1}^{n} \frac{y_j}{k} I(r_j^\pi \le k), \tag{3}$$

$$\text{RBP}(\pi, Y) = (1-p) \sum_{j=1}^{n} y_j p^{r_j^\pi - 1} \tag{4}$$

where $p \in [0, 1]$ is a constant value. y_j in Eq.(3) and Eq.(4) takes a binary value from 0 and 1, which can be transformed from multi-level ratings like LETOR[1].

3. MOTIVATION

For the aggregation task in crowdsourcing, pointwise and pairwise rank aggregation methods will be unsuitable, because both the position importance and annotator quality, which are two main characteristics of crowdsourced data, are not incorporated in these aggregation methods.

First, both the traditional pointwise and pairwise methods define the distance from a local perspective. For example, the Spearman Rank Correlation used in Borda Count [1] treats each object equal, and measures the distance between two ranking lists π and τ as $\sum_{j=1}^{n} (\pi(x_j) - \tau(x_j))^2$, where $\pi(x_j)$ and $\tau(x_j)$ means the position of x_j in the ranking list π and τ respectively. The distances used in pairwise methods treat each object pair as equal, and do not distinguish the different impact of pairs from different positions. Therefore, both pointwise and pairwise methods ignore the position importance in ranking, which is nevertheless critical for rank aggregation. Specifically, one usually cares more about objects ranked high in the output ranking list, and thus objects with different positions as well as pairs constructed from different positions should have different impact on the distance measure.

Second, the reliability of judgments obtained at low cost from crowdsourcing services varies significantly, which is the major difference from traditional aggregation task. Experimental results in [26] shows that the aggregation methods taking annotator quality into consideration, such as MPM [26] always achieve better performances than those without annotator quality factor, such as Borda Count [1] minimizing $\sum_{i=1}^{m} \sum_{j=1}^{n} (\pi(x_j) - \tau_i(x_j))^2$. So it is reasonable to satisfy the majority for some cases while put emphasis on a small subset for the other cases, which can be adjusted by weighting judgments from various sources differently.

To tackle these challenges, we propose to define the distance in a listwise way which taking both position importance and annotator quality into consideration, namely listwise rank aggregation approach.

[1] http://research.microsoft.com/en-us/um/beijing/projects/letor/

4. LISTWISE RANK AGGREGATION

Inspired by the fact that position importance is considered in IR evaluation measure such as NDCG and RBP, we propose to directly utilize these measures as the distance to optimize. Specifically, the output ranking list is viewed as (unknown) ground-truth, and the input judgments from annotator i can be viewed as some observations of the ground-truth waiting for evaluation. When defining the specific distance based on these evaluation measures, however, we find the characteristics of crowdsourcing pose great challenges to this approach.

Firstly, the incompleteness and labeling form (ratings or preferences) make the computation of these measures impossible. As we described above, the annotators usually only select a subset of data for labeling, thus the labeled data are incomplete. Furthermore, labels from annotators either in ratings or preferences, are quite different from the required full-order ranking input in the computation of evaluation measures. Therefore, the direct computation of evaluation measures based on these kinds of data is not available.

Secondly, the annotator quality is not included in these evaluation measures. As we know, the reliability of annotators can vary significantly in crowdsourcing, which make annotator quality an important factor to consider in rank aggregation. However, it is not easy to include this factor in the computation of these evaluation measures.

To address these challenges, we propose to map the judgments (ratings or preferences) to input ranking, and incorporate annotator quality during this process. In the following subsection, we will introduce how we conduct the mapping and incorporation process.

4.1 Listwise Mapping

We cannot induce a ranking list directly by sorting based on ratings or pairwise comparisons mainly due to their incompleteness. In order to make the computation of evaluation measures possible with the input judgments (ratings or preferences), we propose a mapping function from the judgments to input rankings. By analyzing these evaluation measures, we find that the rank of an object is actually needed in the computation. Therefore, we turn to the problem how to map the judgments to obtain the rank of each object. However, since the provided judgments are often incomplete, we incorporate uncertainty into the mapping process. Specifically, we view the pairwise contest as a random variable, then the rank of each object can be defined based on all the results of pairwise contests. Furthermore, the annotator quality can be incorporated in the derivation of the rank distribution.

4.1.1 Randomized Pairwise Comparisons

A pairwise contest between x_s and x_t refers to deciding which one is ranked higher in terms of annotator i. We view each pairwise contest as a Bernoulli trial, so that the result that x_s wins the contest denoted as X_{st} follows the Bernoulli distribution, $X_{st}^{\tau_i} \sim Binomial(1, P(x_s \prec_{\tau_i} x_t))$, where pairwise probability $P(X_{st}^{\tau_i} = 1) = P(x_s \prec_{\tau_i} x_t)$ means the probability that x_s is ranked higher than x_t, denoted as $p_{st}^{\tau_i}$.

To estimate the pairwise probability $p_{st}^{\tau_i}$ when the input judgments are ratings, we first transform the rating data to preference data so that we can treat both types of judgements in a unified way.

$$\tau_i(x_s, x_t) = \begin{cases} 1, & \text{if } \tau_i(x_s) > \tau_i(x_t), \\ 0, & \text{if } \tau_i(x_s) < \tau_i(x_t), \\ Null, & \text{otherwise.} \end{cases} \quad (5)$$

According to previous studies [8, 20], the pairwise probability that one document x_s is more relevant than x_t is dependent on the relative rank position difference between two documents. The basic idea is to estimate this probability by leveraging all pairwise preference relationships conveyed in judgments from an annotator. On the basis of the preference data (either directly provided in preference judgments or generated from ratings as above), we give the estimation of the pairwise probability $p_{st}^{\tau_i}$ as shown in Eq.(6).

$$p_{st}^{\tau_i} = \begin{cases} \frac{N_i(x_s) - N_i(x_t)}{n}, & N_i(x_s) > N_i(x_t), \\ 1 - \frac{N_i(x_t) - N_i(x_s)}{n}, & N_i(x_s) < N_i(x_t), \\ 0.5, & N_i(x_s) = N_i(x_t) \text{ or } \tau_i(x_s, x_t) = Null, \end{cases} \quad (6)$$

where $N_i(x_j)$ denotes the number of pairwise contests that x_j wins in τ_i, which is determined by the known pairwise preferences in Eq.(7),

$$N_i(x_j) = \sum_{l=1, l \neq j, \tau_i(x_j, x_l) \neq Null}^{n} \tau_i(x_j, x_l). \quad (7)$$

4.1.2 Incorporating Annotator Quality

To incorporate annotator quality into the distance, we propose to introduce a parameter η_i to stand for the quality of annotator i. It is natural that we define the parameter as the degree the judgments by annotator i agrees with the (unknown) ground-truth. Since we have turned both types of input judgements into preference data above, we define the degree based on preferences in a unified way. That is the ratio of the number of preference pairs appear in both the judgments of annotator i and ground-truth against the number of all the preference pairs in ground-truth. The formal definition of the quality of annotator i is represented as follows.

$$\eta_i = \frac{\sum_{s,t} I(\tau^*(x_s, x_t) = 1 \&\& \tau_i(x_s, x_t) = 1)}{\sum_{s,t} I(\tau_i(x_s, x_t) = 1)}, \quad (8)$$

where τ^* denotes the ground-truth and $I(\cdot)$ is the indicator function. We can see that when annotator i is perfect, we have $\eta_i \approx 1$; if he/she is a random spammer, we have $\eta_i \approx 0.5$; if he/she is a malicious (i.e. poorly informed) spammer, we have $\eta_i \approx 0$. Note that in estimation of the annotator quality, the ground-truth τ^* is actually unknown. Instead, we estimate the annotator quality in an iterative way by taking the aggregated ranking in previous step as the approximation of the ground-truth. Please refer to section 5.3 for detailed machinery to parameter estimation.

With the annotator quality defined above, we modify the pairwise probability $p_{st}^{\tau_i}$ by incorporating this quality factor as follows

$$\eta_i p_{st}^{\tau_i} + (1 - \eta_i)(1 - p_{st}^{\tau_i}). \quad (9)$$

4.1.3 From Pairwise Contests to Rank Distribution

We now derive the rank of an object. When the results of pairwise contests are deterministic and the data is complete,

Algorithm 1 Iterative Procedure for Rank Distribution of x_j in τ_i.

Input: (1)An item set D with n items; (2)Pairwise preferences from input τ_i with Eq.(5); (3)Annotator quality η_i;
Output: A rank distribution of item x_j on n positions.

Compute $\{p_{jl}^{\tau_i}\}_{l=1,l\neq j}^{n}$ using Eq.(9);
initialize the distribution:$[P^{(0)}(r_j^{\tau_i}=0), P^{(0)}(r_j^{\tau_i}=1)] = [1,0]$;
$t=1$;
for each $x_l \in D - x_j$ **do**

$$[P^{(t)}(r_j^{\tau_i}=0),\ldots,P^{(t)}(r_j^{\tau_i}=t+1)] = [P^{(t-1)}(r_j^{\tau_i}=0),\ldots,P^{(t-1)}(r_j^{\tau_i}=t)] * [p_{jl}^{\tau_i}, 1-p_{jl}^{\tau_i}]; \quad (10)$$

$t++$;
end for
return $[P^{(n-1)}(r_j^{\tau}=0),\ldots,P^{(n-1)}(r_j^{\tau}=n-1)]$.

Algorithm 2 Divide-and-Conquer Algorithm for Rank Distribution of x_j in τ_i.

Input: (1)An item set D with n items; (2)Pairwise preferences from input τ_i with Eq.(5); (3)Annotator quality η_i;
Output: A rank distribution of item x_j on n positions $[P(r_j^{\tau_i}=0),\ldots,P(r_j^{\tau_i}=n-1)]$.

Compute $\{p_{jl}^{\tau_i}\}_{l=1,l\neq j}^{n}$ using Eq.(9) and $p_{jj}=1$;
return DC-RankDistribution($\{p_{jl}^{\tau_i}\}_{l=1}^{n}$,1,$n$).

DC-RankDistribution($\{p_{sl}\}_{l=1}^{n}$,a,b):
if $a+1==b$ **then**,
    return $[p_{sa}, 1-p_{sa}] * [p_{sb}, 1-p_{sb}]$;
else if $a==b$ **then**,
    return $[p_{sa}, 1-p_{sa}]$;
end if
mid=(a+b)/2;
return DC-RankDistribution($\{p_{sl}\}_{l=1}^{n}$,a,mid)*DC-RankDistribution($\{p_{sl}\}_{l=1}^{n}$,mid + 1,b);

the rank of an object x_j provided by annotator i is determined by the number of objects being beaten in Eq.(11).

$$r_j^{\tau_i} = \sum_{l=1,l\neq j}^{n} I(x_j \prec_{\tau_i} x_l) \quad (11)$$

When the pairwise contests are random experiments like Bernoulli trials, the rank of object x_j provided by annotator i is a random variable, which means the number of successes of the $n-1$ independent Bernoulli trials defined in Eq.(12).

$$r_j^{\tau_i} = \sum_{l=1,l\neq j}^{n} X_{jl}^{\tau_i} \quad (12)$$

The distribution of $r_j^{\tau_i}$ is the convolution of the individual probability density distributions [21]. This yields an iterative computation of the rank distribution with complexity $O(n^2)$, as shown in Alg. 1. The operator $*$ in Alg. 1 stands for the convolution operation. Specifically, the distribution of $r_j^{\tau_i}$ is initialized ($t=0$) by the indicator function with the probability at the top position is 1 and 0 otherwise. When a new object x_l comes, the rank distribution of x_j is updated by the convolution between its current rank distribution and the pairwise contest distribution denoted as $[p_{jl}^{\tau_i}, 1-p_{jl}^{\tau_i}]$ estimated by annotator τ_i. For the iteration t, the running time of this convolution computation between the rank distribution (a sequence with length $t+2$) and pairwise contest distribution (a sequence with length 2) is $O(2t)$. The final distribution ($t=n-1$) of $r_j^{\tau_i}$ involves such $n-1$ convolutions, so the time complexity for computing rank distribution of x_j is $\sum_{t=0}^{n-1} 2t = \mathcal{O}(n^2)$.

The time complexity of the Alg. 1 can be improved with divide-and-conquer strategy. The main idea comes from the fact that the required rank distribution can be divided into convolutions of two parts: the first part is convolutions of the first $\frac{n}{2}$ objects, and the second part is convolution of the rest $\frac{n}{2}$ objects. For each convolution, the divide-and-conquer process continues. Therefore, we can change the original sequential computation process to the following divide-and-conquer process as shown in Alg. 2. Suppose the distribution of pairwise contest between x_j and x_j is denoted as $[1,0]$, then the rank distribution of x_j by annotator τ_i can be computed as the convolution of n sequences with each sequence represented as a pairwise contest distribution $[p_{jl}^{\tau_i}, 1-p_{jl}^{\tau_i}]$. For the first iteration ($T=1$), there are $\frac{n}{2}$ convolution computations between any two sequences with length 2, thus $\frac{n}{2}$ sequences with length 3 will be obtained; for the T-th iteration, there are $\frac{n}{2^T}$ convolutions between any two sequences with length $2^{T-1}+1$, and thus $\frac{n}{2^T}$ sequences with length 2^T+1 will be obtained by FFT (Fast Fourier Transform). Finally, the rank distribution of x_j will be derived after $\log_2 n$ iterations, and the time complexity for this efficient algorithm is $\sum_{T=1}^{\log_2 n} \frac{n}{2^T}(2^T+1)\log_2(2^T+1) = O(n\log_2^2 n)$.

4.2 Expected Ranking Measures

Through listwise mapping described above, all the judgments (ratings or preferences) provided by each annotator can be transformed to the listwise information, described as ranks of all the objects in D with estimated rank distributions. Therefore, the input data has become the form of ranking lists.

Recall that we view the output consensus ranking as the ground-truth, and the input ranking lists from each annotator as observations waiting for evaluation. In this sense, the computation of traditional evaluation measures is still infeasible in application, since the ground-truth is in the form of a ranking list. Inspired by using extended evaluation measures such as κ-NDCG and κ-RBP for ranking based ground-truth as in [19], where a mapping function $\kappa : \kappa(r_j^{\pi}) = \frac{n-r_j^{\pi}}{n}$ is utilized to map the rank r_j^{π} to ground-truth label, we propose to define the listwise distance on the basis of these measures as follows.

$$\kappa\text{-NDCG}(\pi,\tau_i) = \sum_{j=1}^{n} \frac{g(\kappa(r_j^{\pi}))D(r_j^{\tau_i})}{\text{DCG}_{\max}(n)},$$

$$\kappa\text{-RBP}(\pi,\tau_i) = \sum_{j=1}^{n} \kappa(r_j^{\pi}) p^{r_j^{\tau_i}-1}.$$

Obviously they can be represented as the following general form $Ev(\pi,\tau_i) = \sum_{j=1}^{n} v(r_j^{\pi}, r_j^{\tau_i})$, where Ev stands for any evaluation measures.

The above measures represent a distance between the output consensus ranking π and the input ranking with respect to τ_i. Since the input ranking with respect to τ_i is stochastic according to section 4.1.3, we propose to use expectation as the distance. As a consequence, the expectation over the rank distribution is conducted, and we obtain the expected measures defined as follows as the final listwise distance.

$$Ev_s(\pi,\tau_i) = \sum_{j=1}^{n}\sum_{r=0}^{n-1} v(r_j^{\pi}, r_j^{\tau_i}) P(r_j^{\tau_i}=r) \quad (13)$$

5. OPTIMIZATION ALGORITHMS

In this section, we investigate optimization algorithms to solve the problem maximizing these expected measures, such as κ-NDCG$_s$ and κ-RBP$_s$. Firstly, we find that these expected measures can be represented as a summation of the utility function for each item at a certain position, and obtain the final objective function $\mathcal{L}(\eta, \pi)$ parameterized by annotator quality $\eta = (\eta_1, \cdots, \eta_m)$ and the aggregated ranking π. Then given the annotator quality η, the optimization problem can be easily interpreted as the assignment problem with a cost function in the form of linear sum, called the Linear Sum Assignment Problem (LSAP for short). Therefore we can transform the optimization problem to the LSAP. With each utility function expressed as a special product form in expected measures, the optimal ranking can be directly obtained by Rearrangement Inequality. After the optimal ranking is obtained, we update the annotator quality η with its definition formula, and repeat the above optimization process until the obtained ranking list keeps unchanged. In this way, we obtain an alternating approach to optimize $\mathcal{L}(\eta, \pi)$, referred to as CrowdAgg.

5.1 General Form of Expected Measures

As shown in Eq. (13), these expected ranking measures can be formulated into a general form as the sum of utility functions. Each utility function measures the utility that an object with ranking information encoded in its rank distributions, is placed at some position in the output permutation, denoted as u_s in this paper. As a consequence, the original expected measures can be rewritten as follows:

$$Ev_s(\pi, \tau_i) = \sum_{j=1}^{n} u_s(r_j^\pi), \text{where } u_s(r_j^\pi) = \sum_{r=0}^{n-1} v(r_j^\pi, r) P(r_j^\tau = r).$$

The optimization objective is to find a permutation which maximizes the sum of utility functions in Eq. (14).

$$\max_{\pi \in \Pi, \eta \in [0,1]^m} \mathcal{L}(\eta, \pi) = \sum_{i=1}^{m} Ev_s(\pi, \tau_i) = \sum_{j=1}^{n} \sum_{i=1}^{m} u_s(r_j^\pi) \tag{14}$$

5.2 Optimization with fixed Parameter η

Through the representation as the sum of utility functions in the above section, we can easily interpret this optimization problem with fixed annotator quality η as to find the optimal assignment of positions for each object according to inputs $(\tau_1, \cdots, \tau_m)$. Moreover, each utility function is a function on an individual object and its output position. Therefore, the optimization problem in our listwise rank aggregation is intrinsically a Linear Sum Assignment Problem (LSAP), which is formalized as follows.

The object set is denoted as $D = \{x_1, \ldots, x_n\}$ and the possible position set is represented as $R = \{1, \ldots, n\}$. For any object x_j, the permutation π can be viewed as the assignment of object $x_j, j = 1, \ldots, n$ to the position r_j^π, which brings about the utility denoted by $w(r_j^\pi, j) = \sum_{i=1}^{m} u_s(r_j^\pi) = \sum_{i=1}^{m} \sum_{r=0}^{n-1} v(r_j^\pi, r) P(r_j^{\tau_i} = r)$.

To find the optimal permutation, we need to optimize the following function in Eq. (15).

$$\max_{\pi \in \Pi} \sum_{j=1}^{n} w(r_j^\pi, j) \tag{15}$$

Table 1: Factorizations of Expected Ranking Measures

measures	$f(\cdot)$	$h(\cdot)$
κ-NDCG$_s$	$\frac{1}{\text{DCG}_{max}(n)} \sum_{r=0}^{n-1} \frac{P(r_j^{\tau_i}=r)}{\log(1+r)}$	$g(\kappa(r_j^\pi))$
κ-RBP$_s$	$(1-p)\sum_{r=0}^{n-1} p^{r-1} P(r_j^{\tau_i} = r)$	$\kappa(r_j^\pi)$

Many classical algorithms such as Hungarian method [14] have been proposed to solve this LSAP. For a LSAP with size n, the running time of Hungarian algorithm is $O(n^4)$. The high time complexity motivates us to further probe the optimization problem. In light of the factorization in $v(\cdot, \cdot)$ [15], we find that the utility function $u_s(\cdot, \cdot)$ for most expected measures can be factorized as a product of two functions. One is on positions in output permutation denoted as $h(\cdot)$, and the other is the function of position per object in the estimated ranking input denoted as $f(\cdot)$. The mathematical form is shown as below.

$$u_s(r_j^\pi, r_j^{\tau_i}) = h(r_j^\pi) f(r_j^{\tau_i}) \tag{16}$$

For example, κ-NDCG$_s$ and κ-RBP$_s$ can both be represented in this way. Specifically, two different kinds of definition of $f(\cdot)$ and $h(\cdot)$ with regard to these expected measures are listed in Table 1.

According to the definition of $w(r_j^\pi, j)$ and the factorization of utility function $u_s(\cdot, \cdot)$ in Eq. (16), $w(r_j^\pi, j) = \sum_{i=1}^{m} h(r_j^\pi) f(r_j^{\tau_i}) = h(r_j^\pi)(\sum_{i=1}^{m} f(r_j^{\tau_i}))$.

Let $u_j = h(r_j^\pi)$ and $v_j = \sum_{i=1}^{m} f(r_j^{\tau_i})$, the infimum and supremum of the objective function in Eq. (14) will be directly obtained by the Rearrangement Inequality [11] as shown in Lemma 1.

LEMMA 1. *Let $0 \le u_1 \le \ldots \le u_n$ and $0 \le v_1 \le \ldots \le v_n$. Then for any permutation ϕ*

$$\sum_{i=1}^{n} u_i v_{n+1-i} \le \sum_{i=1}^{n} u_i v_{\phi(i)} \le \sum_{i=1}^{n} u_i v_i.$$

Specifically, we can see that $u_j = h(r_j^\pi)$ and $v_j = \sum_{i=1}^{m} f(r_j^{\tau_i})$ satisfies the non-negative conditions since $f(\cdot)$ and $h(\cdot)$ are both positive. Sorting $x_1, \ldots, x_n$ according to v_j in descending order, we obtain the optimal permutation π_G with the supremum of the objective function based on Lemma 1. As a consequence, we obtain a simple algorithm which can efficiently find the optimal solution for the optimization problem with fixed η by sorting based on v_j. The details of algorithm are shown in Alg. 3.

Algorithm 3 Maximize $\mathcal{L}(\eta, \pi)$ with fixed η

Input: An item set D and a collection of ranking inputs $\{\tau_1, \ldots, \tau_m\}$ over D;
Output: An aggregated ranking π.
1: Compute $F(x_j) = \sum_{i=1}^{m} f(r_j^{\tau_i})$, where f is defined for different measures in Table 1;
2: Obtain the permutation π by sorting $F(x_j)$ in descending order;
3: **return** π.

5.3 Estimating Parameter η with Fixed π

Recall that η is defined as the degree the judgments by annotator i agrees with the unknown ground-truth. We propose an iterative way to estimate η by taking the currently aggregated ranking list π as the approximation of the ground-truth. Therefore, η can be estimated, where each η_i

is calculated as the pairwise agreement between π and the input τ_i.

$$\eta_i = \frac{\sum_{s,t} I(\pi(x_s, x_t) = 1 \&\& \tau_i(x_s, x_t) = 1)}{\sum_{s,t} I(\tau_i(x_s, x_t) = 1)} \quad (17)$$

5.4 CrowdAgg

As described above, a natural optimization strategy to maximize the objective $\mathcal{L}(\eta, \pi)$ is the alternating approach: (1) initialize $\eta = 1$; (2) fix η and optimize over π by LSAP as described in Alg. 3; (3) fix π and estimate the η as described in Eq. (17); (4) repeat step (2) and (3) until the obtained permutations keep stable. This alternating approach for optimizing $\mathcal{L}(\eta, \pi)$ is referred to as CrowdAgg.

6. EXPERIMENTS

In this section we present our experimental results. Firstly, we introduce the experimental setting, including data sets, baseline methods and evaluation measures. Then we present ranking performance comparison between our proposed CrowdAgg and the baseline methods. In addition, we empirically study the annotator accuracy and the robustness of these rank aggregation methods with respect to two kinds of noisy annotators, such as spammers and malicious annotators [4]. Finally, we compare the running time of the proposed CrowdAgg with the state-of-the-art rank aggregation algorithms.

6.1 Experimental Setting

Here we implement our listwise rank aggregation methods CrowdAgg where Alg. 2 is used to compute rank distributions. We denote the algorithm optimizing expected measures κ-NDCG and κ-RBP as CrowdAgg$_{\text{NDCG}}$ and CrowdAgg$_{\text{RBP}}$, respectively.

Data Sets. To evaluate the performance of our proposed method, we use two data sets from two different kinds of crowdsourcing tasks as described below. Conventional rank aggregation data sets like Million Query data sets from meta-search application may not simulate the crowdsourced setting well, so we utilize such two benchmark data sets directly from crowdsourced labeling tasks.

(1) *Graded Judgment Data from Crowdsourced Relevance Labeling Task. CS-TREC2011* data set is a collection of topic-document pairs labeled as relevant or non-relevant by Mechanical Turks. Therefore, the judgments are provided in the form of ratings. There are 100 topics with 190 documents per topic, and the number of annotators is varied with topics, about 43 on average. Each annotator rates less than 10 documents on average.

(2) *Preference Judgment Data from Crowdsourced Similarity Labeling Task. MIREX2005* data set is a collection of music form Symbolic Melodic Similarity task in Music Information Retrieval Evaluation eXchange (MIREX) in 2005. This task aims to find a ranking list of musical pieces according to the similarity to certain piece of music, which acts as a query in information retrieval. There are 11 pieces of music as queries. Pairwise preference labels are collected from Mechanical Turks in [25]. The ground-truth is in the form of 5-grade judgements [24].

Baseline Methods. Since pointwise methods and listwise methods can only be applied to graded judgment data[2], we use pairwise rank aggregation methods as our baselines to obtain a comprehensive comparison on both data sets. According to whether annotator quality is considered in the model, traditional pairwise rank aggregation methods fall into two categories: (1) Methods without annotator quality, such as CondorcetFuse [18], GreedyOrder [5], BradleyTerry [23]; (2) Methods with annotator quality, such as MPM [26], Crowd-BT [4] and EMEloRating [2, 27]. Here we describe some parameter settings in these methods: the learning rates used in gradient methods for BradleyTerry (0.1 for both data sets), MPM (0.1 for MusicCrowd and 0.0001 for TREC2011) and Crowd-BT (0.01 for both data sets) were chosen according to the best performance in terms of NDCG@10, and the parameter setting in EMEloRating is the same as in [2].

Evaluation Methods. There are several routines to evaluate these unsupervised rank aggregation methods. One of the leading methods is to use traditional IR evaluation measures with ground-truth labels given for evaluation. In this paper we use NDCG@$\{3, 5, 7, 9, 10\}$, RBP, Precision@$\{3, 5, 7, 9, 10\}$ and MAP for evaluation. Routinely the constant value p in RBP is 0.95. In MIREX2005, the transformation of five-grade relevance labels to binary values is mentioned in the background for RBP.

6.2 Ranking Accuracy Analysis

The ranking performance comparison results on both CS-TREC2011 and MIREX2005 are shown in Table 2. It is clear that our proposed methods CrowdAgg outperform the baseline methods on both data sets in most cases.

In terms of NDCG@10, CrowdAgg$_{\text{NDCG}}$ achieves 1.5% improvement with respect to the best pairwise rank aggregation method, i.e. GreedyOrder on CS-TREC2011, and 6.2% improvement with respect to the best pairwise method, i.e. MPM on MIREX2005. In terms of RBP, the improvement of CrowdAgg$_{\text{RBP}}$ is 5.6% with respect to the best pairwise method, i.e. GreedyOrder on CS-TREC2011, and 5.4% with respect to the best pairwise method.

The improvements can be explained as follows: (1) Compared with the first kind of pairwise rank aggregation methods (without annotator quality), CrowdAgg are superior by distinguishing the reliability of preference judgments by the annotator quality; (2) Compared with the second kind of pairwise rank aggregation methods, CrowdAgg obtain better ranking performance by treating the judgments per annotator in a listwise way and taking into consideration of position importance in optimization. Besides, more precise estimation of annotator quality η can be obtained from better aggregated ranking and enhance each other in an alternating way in CrowdAgg.

6.3 Annotator Quality Analysis

In this section, we empirically study the influence of annotator quality. Firstly, we validate the hypothesis that annotator quality will impact much to the results and thus should be taken into consideration in the rank aggregation method. Secondly, we compare the quality distribution among different methods as conducted in [4].

6.3.1 Impact of Annotator Quality

To validate the hypothesis that annotator quality is important for rank aggregation, we propose to verify whether the aggregation methods with annotator quality will perfor-

[2] St.Agg [20] does not incorporate the annotator quality.

Table 2: Ranking Performance Comparison of Preference Aggregation Methods on two data sets (CS-TREC2011 and MIREX2005). For each specific metric, the result with bold type is significantly better than the other corresponding results through one-tailed and paired t-tests (p-value< 0.05).

(a) CS-TREC2011

Methods	P@3	P@5	P@7	P@9	P@10	MAP	N@3	N@5	N@7	N@9	N@10	RBP
CondorcetFuse	0.597	0.604	0.621	0.626	0.621	0.560	0.591	0.597	0.609	0.613	0.611	0.564
GreedyOrder	0.710	**0.698**	0.674	0.661	0.657	0.604	0.713	**0.704**	0.688	0.679	0.676	0.621
BradleyTerry	0.373	0.370	0.386	0.378	0.377	0.326	0.377	0.375	0.384	0.379	0.379	0.345
MPM	0.690	0.690	0.671	0.663	0.657	0.570	0.687	0.687	0.676	0.671	0.667	0.580
Crowd-BT	0.403	0.416	0.416	0.422	0.417	0.534	0.404	0.412	0.413	0.417	0.414	0.473
EMEloRating	0.690	0.666	0.673	0.654	0.649	0.588	0.691	0.676	0.679	0.667	0.663	0.602
CrowdAgg$_{\text{NDCG}}$	**0.713**	0.694	0.680	0.673	0.673	0.635	**0.715**	0.702	**0.692**	**0.687**	**0.686**	0.657
CrowdAgg$_{\text{RBP}}$	0.680	0.684	**0.686**	**0.679**	**0.674**	**0.636**	0.683	0.686	0.686	0.682	0.679	**0.658**

(b) MIREX2005

Methods	P@3	P@5	P@7	P@9	P@10	MAP	N@3	N@5	N@7	N@9	N@10	RBP
CondorcetFuse	0.364	0.455	0.468	0.434	0.418	0.472	0.523	0.596	0.514	0.478	0.414	0.170
GreedyOrder	0.273	0.418	0.455	0.424	0.409	0.459	0.444	0.525	0.490	0.462	0.399	0.171
BradleyTerry	0.242	0.400	0.429	0.424	0.499	0.441	0.412	0.496	0.460	0.447	0.385	0.170
MPM	0.485	0.509	0.494	0.475	0.446	0.516	0.695	0.693	0.612	0.576	0.501	0.179
Crowd-BT	0.576	0.564	0.507	0.465	0.446	0.608	0.814	0.801	0.640	0.578	0.496	0.200
EMEloRating	0.364	0.436	0.455	0.424	0.409	0.495	0.606	0.629	0.568	0.511	0.436	0.203
CrowdAgg$_{\text{NDCG}}$	**0.697**	**0.600**	**0.546**	**0.485**	**0.455**	0.712	**0.955**	**0.880**	**0.703**	**0.621**	**0.532**	0.214
CrowdAgg$_{\text{RBP}}$	0.697	0.600	0.546	0.485	0.455	**0.713**	0.955	0.879	0.702	0.620	0.531	0.**215**

m better than that without annotator quality. Therefore, we consider the methods with annotator quality, such as MPM, Crowd-BT, EMEloRating and our proposed CrowdAgg$_{\text{NDCG}}$. These methods can be easily reduced to the version without annotator quality by treating each annotator equally. Specifically, we set these annotator quality parameters $\eta = 1$ as in MPM, Crowd-BT and CrowdAgg$_{\text{NDCG}}$, and denote the new corresponding version as MPM-eq, Crowd-BT-eq, CrowdAgg$_{\text{NDCG}}$-eq. While for EMEloRating, we simply take all the pairwise preferences as output to the Elo rating system without filtering by annotator quality, denoted as EMEloRating-eq.

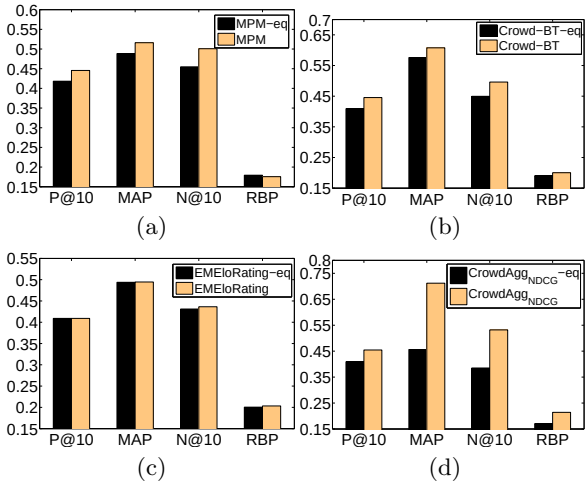


Figure 1: Ranking Performance Comparison between models without and with annotator quality on MIREX2005 (The comparison results are significant through paired t-tests with p-value< 0.05.)

We conduct experiments on MIREX2005 with the two groups of aggregation methods, as mentioned above. The ranking performances are compared between the original one and the corresponding reduced version in terms of P@10, MAP, NDCG@10 and RBP. The comparison results are shown in Fig. 1. Obviously the method with annotator quality can significantly outperform the corresponding method without annotator quality. For example, the improvement of Crowd-BT is 10.3% with the consideration of annotator quality in terms of NDCG@10. The largest performance difference between CrowdAgg$_{\text{NDCG}}$ and CrowdAgg$_{\text{NDCG}}$-eq is 56% with respect to MAP, while the smallest performance difference is 11.1% in terms of P@10. Note that the performance difference between EMEloRating and EMEloRating-eq is relatively small. Through our analysis, we find the major reason lies in the EM process of the EMEloRating-eq method, which also takes the annotator quality into consideration in essence. In other words, EMEloRating-eq is not a true reduced version without annotator quality.

6.3.2 *Accuracy for Different Distributions of Annotator Quality*

Here we study the difference between the estimated annotator quality distribution and the ground-truth quality distribution. We say the estimated annotator quality is more accurate if the difference is small. There are various definition of annotator quality for different aggregation methods, such as adherence parameter in MPM [26], pairwise preference accuracy ACC in Crowd-BT [4] (known as the accuracy based on Wilcoxon-Mann-Whitney statistics employed in [26]), and confusion matrix C in EMEloRating. In this paper, we conduct a similar empirical study as in [4], and thus employ pairwise preference accuracy ACC to compute the annotator quality.

$$ACC = \frac{\sum_{s,t} I(y_s > y_t) I(\pi(x_s, x_t))}{\sum_{s,t} I(y_s > y_t)}, \tag{18}$$

where y_s is the ground-truth label for each item $x_s \in D$.

Specifically, we conduct experiments on MIREX2005 with 11 topics to evaluate the quality distribution of annotators for each topic. We calculate the KL-divergence between the estimated quality distribution and the ground-truth distribution. The average divergence score is for CrowdAgg$_{\text{NDCG}}$, Crowd-BT, EMEloRating and MPM is 1.43, 2.86, 2.14,1.93, respectively. Obviously, CrowdAgg$_{\text{NDCG}}$ achieves the smallest divergence. It indicates that our proposed CrowdAgg can make better estimation on the annotator quality than the baseline methods. We randomly selected 9 topics from MIREX2005 and plot the quality distributions of different methods and groud-truth in Fig. 2 for illustration. Each sub figure is corresponding to one topic. From Fig. 2 we can al-

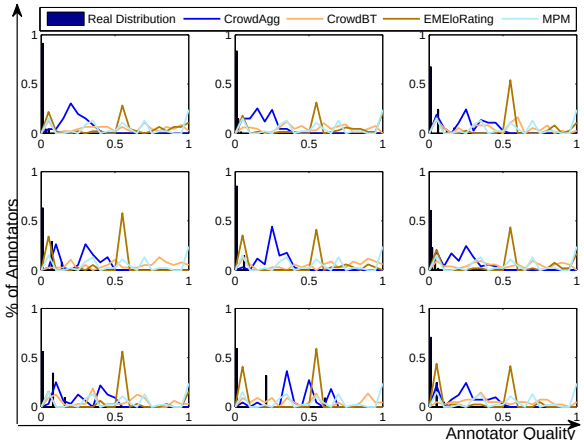


Figure 2: Quality Distribution of Annotators computed from the ground-truth labels (Real Distribution) and Estimated in aggregation methods on randomly sampled 9 topics of MIREX2005

so find that the distribution estimated from CrowdAgg$_{\text{NDCG}}$ (i.e. blue curve) in each sub figure is more similar to the real distribution (i.e. black histogram) than other methods.

6.4 Robustness Analysis

In this section we investigate the robustness of aggregation algorithms to spammers. Similar to [4], we mainly care about two kinds of spammers, random spammers and malicious (i.e. poorly informed) spammers. The random spammers assign the label randomly, while the malicious spammers assign the wrong label most of time. We investigate the robustness of all these aggregation methods to two kinds of spammers respectively, and the experimental results are shown only on MIREX2005 for space limitation. Similar results can be also obtained on CS-TREC2011.

6.4.1 Random Spammers

To simulate the labeling behavior of spammers, we define the decision function on the labels of the pairwise objects between x_i and x_j. For random spammer, the decision function τ_{RandSpam} as follows is proposed, where $y(x_i, x_j)$ is the preference label inferred from the ground-truth relevance labels in MIREX2005.

$$\tau_{\text{RandSpam}}(x_i, x_j) = \begin{cases} 1 - y(x_i, x_j) \text{ with probability } 0.5 \\ y(x_i, x_j) \text{ with probability } 0.5 \end{cases}$$

Here we consider the robustness of aggregation methods with the addition of random spammers. Since the number of annotators m is different for different topics, we consider the number of random spammers to be added is in proportion to m, and refer the proportion as *Ratio of Added Random Spammers*. We vary the ratio from 0 to 1 with a step of 0.05, and obtain 20 different data sets for each given ratio. The performance is then averaged over the 20 data sets for each ratio. The ranking performance comparisons over different algorithms in terms of P@10, MAP, NDCG@10 and RBP are shown in Fig. 3.

To quantitatively compare the robustness of different methods, we use the coefficient of variation of performance[3], which is a widely used measure for variance comparison. From the results, we find that our proposed CrowdAgg$_{\text{NDCG}}$ achieve the best coefficient of variation in terms of RBP among methods from top to bottom in the legend (0.0227, 0.0245, 0.0436, 0.0330, 0.0312, 0.0191, 0.0185), and is the third best in terms of NDCG@10 among methods from top to bottom in the legend (0.0147, 0.0174, 0.3347, 0.0228, 0.0406, 0.3238, 0.0216). Furthermore, we can see that with the increase of the ratio of added random spammers, CrowdAgg$_{\text{NDCG}}$ can almost always outperform all the other methods. Therefore, we conclude that our proposed CrowdAgg is robust to random spammers.

[3]Coefficient of variation is defined as the the ratio of the standard deviation to the mean

6.4.2 Malicious Spammers

To simulate the labeling behavior of malicious spammers, we define the decision function $\tau_{\text{MaliSpam}} = 1 - y(x_i, x_j)$, where $y(x_i, x_j)$ is the preference label inferred from the ground-truth relevance labels in MIREX2005.

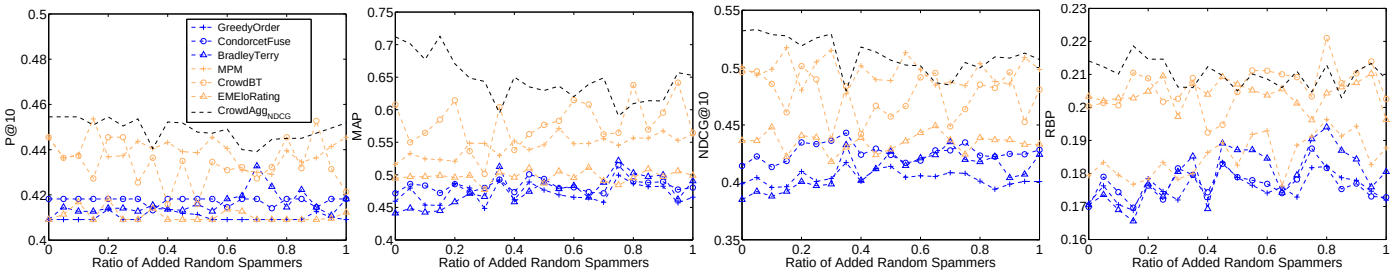
Here we consider the robustness of aggregation algorithms with the addition of malicious spammers. We introduce *Ratio of Added Malicious Spammers* and conduct experiments with different ratios as in the above section. The performance comparison results are depicted in Fig. 4.

Similarly, we also compare the robustness of different methods in terms of coefficient of variation of performance. From the results, we find that our proposed CrowdAgg$_{\text{NDCG}}$ achieve the best coefficient of variation in terms of both NDCG@10 and RBP among methods from top to bottom in the legend (NDCG@10:0.0100, 0.0122, 0.0112, 0.0186, 0.0498, 0.0175, 0.0099; RBP:0.0118, 0.0174, 0.0218, 0.0407, 0.0382, 0.0275, 0.0116). Meanwhile, we can also see that with the increase of the ratio of added malicious spammers, CrowdAgg$_{\text{NDCG}}$ can almost always outperform all the other methods. Therefore, CrowdAgg is also robust to malicious spammers on MIREX2005.

In summary, our proposed CrowdAgg is robust to both random spammers and malicious spammers. Each random spammer always provides half useful information to derive the ground truth ranking, so there is no reason for the performance decline in Fig. 3. Each malicious spammer always provides harmful information to derive the ground truth ranking, so the performance decline in Fig. 4 is deemed to appear when the ratio achieves some threshold. In terms of different spammers, CrowdAgg is more robust to the malicious spammers than the random spammer, which is promising in real application.

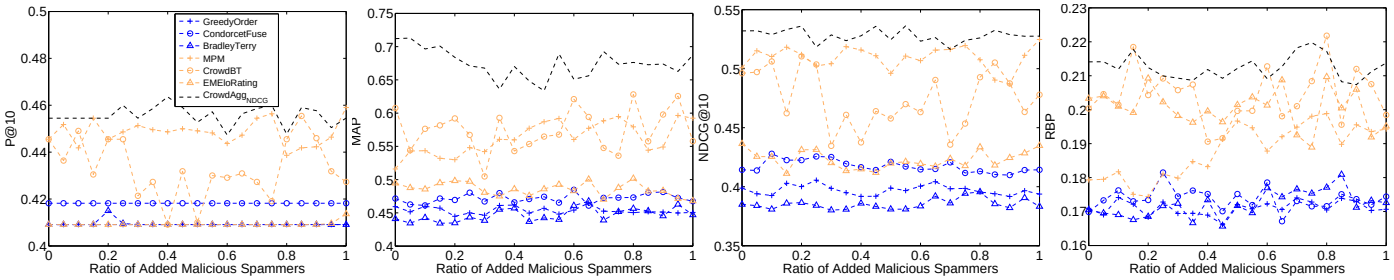
7. CONCLUSION

In this paper, we propose a listwise rank aggregation method in crowdsourcing. The main idea is to adopt IR measures as objective function to take position importance into consideration, as compared with traditional pointwise or pairwise rank aggregation methods. To solve the challenges introduced by the characteristics of crowdsourcing, we propose to map the judgments (ratings or preferences) to input ranking and incorporate annotator quality in this process. So we define the new expected measures, and use them as objective functions. For optimization, we propose a novel alternative optimization algorithm named CrowdAgg based on LSAP and the iterative estimation of annotator quality. Finally, our experimental results on benchmark data sets shows the effectiveness and robustness of our proposed CrowdAgg.



(a) (b) (c) (d)

Figure 3: Ranking Performance Variation along with Ratio of Added Random Spammers on MIREX2005 (Performance variation results for various aggregation methods are significantly different through paired t-tests with p-value< 0.05.)



(a) (b) (c) (d)

Figure 4: Ranking Performance Variation along with Ratio of Added Malicious Spammers on MIREX2005 (Performance variation results for various aggregation methods are significantly different through paired t-tests with p-value< 0.05.)

For future work, we will investigate how to adapt our list-wise rank aggregation method to an active learning setting, which is suitable to be blended to crowdsourcing.

Acknowledgments

This research work was funded by the 973 Program of China under Grants No. 2012CB316303, No. 2014CB340401, the 863 Program of China under Grants No. 2012AA011003, the National Natural Science of China under Grant No. 61472401, No. 61203298, No. 61100072, and the National Key Technology R&D Program of China under Grants No. 2012BAH39B02, No. 2012BAH46B04.

8. REFERENCES

[1] J. A. Aslam and M. Montague. Models for metasearch. SIGIR2001, pages 276–284.

[2] M. Bashir, J. Anderton, J. Wu, P. B. Golbus, V. Pavlu, and J. A. Aslam. A document rating system for preference judgements. SIGIR '13, pages 909–912.

[3] B. Carterette, P. N. Bennett, D. M. Chickering, and S. T. Dumais. Here or there: Preference judgments for relevance. ECIR'08, pages 16–27, 2008.

[4] X. Chen, P. N. Bennett, K. Collins-Thompson, and E. Horvitz. Pairwise ranking aggregation in a crowdsourced setting. WSDM '13, pages 193–202.

[5] W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *JAIR1999*, 10(1):243–270, May.

[6] C. D. Manning, P. Raghavan, and H. Schĺźtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[7] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. pages 301–312.

[8] M. Farah and D. Vanderpooten. An outranking approach for rank aggregation in information retrieval. SIGIR2007, pages 591–598.

[9] D. F. Gleich and L.-h. Lim. Rank aggregation via nuclear norm minimization. KDD2011, pages 60–68.

[10] J. Guiver and E. Snelson. Bayesian inference for plackett-luce ranking models. ICML2009, pages 377–384.

[11] G. H. Hardy, J. E. Littlewood, and G. PÍőlya. *Inequalities*. Cambridge University Press, 1952.

[12] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM TOIS2002*, 20(4):422–446.

[13] J. Kekäläinen. Binary and graded relevance in ir evaluations-comparison of the effects on ranking of ir systems. *IPM*, 41(5):1019–1033, Sept. 2005.

[14] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics*, 2(1-2):83–97, 1955.

[15] Q. Le and A. Smola. Direct optimization of ranking measures. *arXiv Preprint arXiv: 0704.3359*, 2007.

[16] G. Lebanon and J. D. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. ICML2002, pages 363–370.

[17] A. Moffat and J. Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM TOIS2008*, 27(1):2:1–2:27, Dec.

[18] M. Montague and J. A. Aslam. Condorcet fusion for improved retrieval. pages 538–548.

[19] S. Niu, J. Guo, Y. Lan, and X. Cheng. Top-k learning to rank: labeling, ranking and evaluation. SIGIR '12, pages 751–760.

[20] S. Niu, Y. Lan, J. Guo, and X. Cheng. Stochastic rank aggregation. UAI2013, pages 478–487.

[21] A. Papoulis. *Random Variables and Stochastic Processes*. McGraw-Hill, 1991.

[22] P. R.L. The analysis of permutations. *Applied Statistics*, 24(2):193–202, 1974.

[23] L. L. Thurstone. The method of paired comparisons for social values. *The Journal of Abnormal and Social Psychology*, 21(4):384, 1927.

[24] R. Typke, M. den Hoed, J. de Nooijer, F. Wiering, and R. C. Veltkamp. A ground truth for half a million musical incipits. *JDIM*, 3(1):34–38, 2005.

[25] J. Urbano, J. Morato, M. Marrero, and D. Martín. Crowdsourcing preference judgments for evaluation of music similarity tasks. In *SIGIR CSE*, pages 9–16.

[26] M. N. Volkovs and R. S. Zemel. A flexible generative model for preference aggregation. WWW2012, pages 479–488.

[27] J. Wu. Applying em to compute document relevance from crowdsourced pair preferences. Master's thesis, Northeastern University, 2013.