Rank Aggregation for Similar Items

D. Sculley *
Department of Computer Science
Tufts University, Medford, MA, USA
dsculley@cs.tufts.edu

Abstract

The problem of combining the ranked preferences of many experts is an old and surprisingly deep problem that has gained renewed importance in many machine learning, data mining, and information retrieval applications. Effective rank aggregation becomes difficult in real-world situations in which the rankings are noisy, incomplete, or even disjoint. We address these difficulties by extending several standard methods of rank aggregation to consider similarity between items in the various ranked lists, in addition to their rankings. The intuition is that similar items should receive similar rankings, given an appropriate measure of similarity for the domain of interest. In this paper, we propose several algorithms for merging ranked lists of items with defined similarity. We establish evaluation criteria for these algorithms by extending previous definitions of distance between ranked lists to include the role of similarity between items. Finally, we test these new methods on both synthetic and real-world data, including data from an application in keywords expansion for sponsored search advertisers. Our results show that incorporating similarity knowledge within rank aggregation can significantly improve the performance of several standard rank aggregation methods, especially when used with noisy, incomplete, or disjoint rankings.

1 Introduction.

Rank aggregation, the problem of combining the ranked preferences of many experts, has been studied for several centuries and was first driven by the need to design fair elections. This study revealed a number of surprising complexities, including the so-called Condorcet paradox that a candidate who wins every pairwise majority contests is not guaranteed to be the winner of many intuitive election systems [15]. More recently, it was shown that producing an optimal aggregation of even four ranked lists is NP-hard under certain reasonable assumptions [3].

Rank aggregation is an extremely useful tool for modern data mining, especially for handling noisy data. Rank aggregation can be thought of as the unsupervised analog to regression, in which the goal is to find an aggregate ranking that minimizes the distance to each of the ranked lists in the input set. Rank aggregation has also been proposed as an effective method for nearest-neighbor ranking of categorical data [4], and gives a robust approach to the problem of combining the opinions of experts with different scoring schemes, as are common in ensemble methods.

In real world problems, it is often the case that the various experts provide noisy, incomplete rankings. Data mining on information drawn from the Internet is fraught with noise from inconsistent user behavior, malicious "bots", or mistaken user identification. Incomplete rankings may come in two forms. In *top-k* lists, the experts may each provide rankings for only the k best items (such as a ranked list of "Top Ten Movies of All Time"). In *partial* lists, the experts may provide complete rankings over a limited subset of the possible items, due to incomplete knowledge of the item universe (for example, a ranked list of "Movies I've Seen This Month.") Both types of incomplete rankings are common in many real world applications, and each requires different assumptions on the part of the rank aggregation method chosen. Indeed, our work in this area was initially motivated by the need to aggregate ranked lists of keywords for the problem of keywords expansion in sponsored search. As described in the experimental section of this paper, we found that the ranked lists of keywords from many sources were noisy and incomplete, reducing the effectiveness of standard rank aggregation methods.

In this paper, we address the problem of aggregating noisy, incomplete ranked lists through the addition of *similarity* information. The intuition driving this approach is that similar items should be ranked similarly, given an appropriate similarity measure for the data. Thus, similarity can be used to help combat noisy rankings, increase the effectiveness of comparisons between incomplete rankings, and enable rank aggregation of even disjoint lists.

*Work performed at Yahoo!, Inc., in Spring of 2006.

As a motiving example, consider the following toy problem, consisting of ranked lists from two experts:

Expert 1: A, B, C
Expert 2: C', D, E

In these lists, the items **C** and **C'** are highly similar, but are not exact matches for one another. However, if we cannot consider similarity, then the two lists are completely disjoint. If we have no *a priori* reason to prefer the opinions of one expert over another, then standard methods of rank aggregation will interleave the rankings in one of the following ways:

Aggregation 1: A, C', B, D, C, E
Aggregation 2: C', A, D, B, E, C

The first of these aggregations is unsatisfactory, as highly similar items **C** and **C'** are given divergent rankings. In the second example the similar items are given the most divergent rankings possible. However, both the Spearman footrule distance and the Kendall tau distance (two standard methods of measuring distance between ranked lists) will judge both Aggregation 1 and Aggregation 2 to be optimal rankings. Thus, if we are to capitalize on similarity in rank aggregation, we need both new aggregation methods and new evaluation measures.

Accordingly, the methods that we propose in this paper capitalize on the additional information provided by a defined similarity measure. Consider the following alternative, based on rank aggregation with similarity:

Aggregation 3: A, B, C', C, D, E

Aggregation 3 agrees with our intuition that **C'** should be ranked behind both **A** and **B**, while **C** should be ranked ahead of both **D** and **E**. The evaluation methods that we propose in this paper will prefer this third aggregation to either of the previous two.

Contributions. In this paper, we extend previous methods of rank aggregation to capitalize on the benefit of similarity information between ranked items. The rank aggregation methods we extend include Borda's method [15], several Markov Chain methods [3], and the median rank aggregation method [4], each of which may be best suited to different applications. We also extend previous definitions of distance between ranked lists, which enable the evaluation of the effectiveness of these methods. The methods are evaluated empirically on both synthetic and real-world data.

2 Definitions

Before moving further, we formalize the notions of *rankings* and *similarity*.

2.1 Ranked lists. We start with a universe U of items, each with a unique identifier $i \in U$. A ranked list r of items $i_1 \dots i_n$ drawn from a universe U of items is an ordered subset $S \subseteq U$ with each $i \in S$, such that $r = [i_1 \geq i_2 \geq \dots \geq i_n]$. (Here, we draw heavily on the notation of of [3].) For each item $i \in S$, $r(i)$ shows the ranking of item i in the ranked list r. Note that the optimal ranking of any item is 1, rankings are always positive, and higher rank shows lower preference in the list. (We will use the terms *list* and *ranked list* interchangeably.) The number of items in r is given by $|r|$, and we assume that the items in U are each assigned a unique identifier. We refer to the set of items in r by S_r, although we will occasionally refer to items in a list r with the shorthand notation $i \in r$, which should be clear by context. Finally, we define the notation r^n to refer to the item i in r such that $r(i) = n$.

There are several special cases of ranked lists to consider. A *complete ranking* is a list r that contains a ranking for every item $i \in U$. A *partial ranking* is a list r that contains rankings for only a subset of items $S \subset U$. A *Top-k* list is a list r with $|r| = k$, and all other items $i \notin r$ are assumed to be ranked below every item in r by the given expert [5].

A *projection* operator is used in computing distances between partial and complete rankings [3]. When T is a subset of U, and r is a (possibly partial) ranking of items $S_r \subseteq U$, then the projection $r_{|T}$ contains only items $S \cap T$ while keeping the relative orderings from r invariant. The notation $r_{|r_2}$ denotes the projection of r onto the set of objects in S_{r_2}.

2.2 Similarity functions. A pairwise *similarity* function $s(\cdot, \cdot)$ between items in U must satisfy the following requirements. First, the function must be non-negative for all pairwise comparisons: $\forall i, j \in U : s(i, j) \geq 0$. Second, the function must be symmetric: $\forall i, j \in U : s(i, j) = s(j, i)$. Third, the similarity between and item and itself must be greater than zero: $\forall i \in U : s(i, i) > 0$. Finally, the maximum similarity between an item and any other item may not be greater than the similarity between the item and itself: $\forall i, j \in U : s(i, i) \geq s(i, j)$. It will often be the case that a *normalized* similarity function is useful. In these cases, $\forall i \in U : s(i, i) = 1$. One method of normalizing similarity functions is: $s_{norm}(i, j) = \frac{s(i,j)}{\sqrt{s(i,i)s(j,j)}}$.

These similarity requirements are flexible, allowing the use of a wide range of similarity functions over vari-

ety of data types. These include similarity measures on vectors, strings, documents, trees. Other valid similarity measures include the vast array of positive definite *kernels* from the literature on pattern recognition and support vector machines [13] [14]. Formal distance metrics of the form $d(\cdot,\cdot)$ defined in a metric space, such as Euclidean distance, may be transformed into similarity measures through a variety of methods, including the Gaussian kernel $s(i,j) = e^{\frac{-d(i,j)}{\sigma^2}}$ [14].

For convenience, we define a special similarity measure called the *uniqueness function*, $s_0(\cdot,\cdot)$:

$$\forall i,j \in U : s_0(i,j) = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

The uniqueness function (a form of Dirac's delta function) describes the case where every item in U is considered to be completely distinct from all other items in U. We will use this function to show that standard rank aggregation methods are special cases of our extended aggregation methods using similarity.

Our distance measures between partial ranked lists using similarity information will require a *lambda-similarity projection*. This projection is defined from a list r of items $S \subseteq U$ onto a set $T \subseteq U$, and uses a given similarity function $s(\cdot,\cdot)$. The projection yields a new list $r_{\lambda|T}$ that contains a rank-invariant ordering of all the items $j \in T$ for which there exists an item $i \in S$ such that $s(i,j) > \lambda$. Here, λ represents a minimum similarity threshold that i and j must meet for the similarity to be considered meaningful. (In our experiments, we will use $\lambda = 0$.) A possible variant is to include only the d most similar items $j \in T$ for each item $i \in S$, which may provide increased computational efficiency when used in conjunction with data structures for fast nearest neighbor retrieval [10].

3 Measuring Disagreement Among Rankings

On an informal level, we desire that any method of rank aggregation should seek to produce an aggregate ranking with minimal total disagreement among the input lists. In this section, we will extend two previous measures of disagreement between ranked lists, namely the *Spearman footrule distance* and the *Kendall-Tau distance*, to include the case where we have a meaningful similarity measure between items in the lists. This will serve to define formally the problem of rank aggregation with similarity, by establishing a success measure for our methods of merging ranked lists of similar items. Note, however, that not all of these distances will be metric – most notably in the case of partial lists.

3.1 Standard Distance Measures

Here, we review two standard distance measures on ranked lists, the Spearman Footrule Distance and the Kendall Tau distance. We also recall their variants for partial lists, and for groups of lists.

Spearman Footrule Distance. The Spearman Footrule Distance measures the distance between two ranked lists by summing the differences in the rankings of each item. That is, given two complete rankings r_1 and r_2, $F(r_1,r_2) = \sum_{i \in U} |r_1(i) - r_2(i)|$ [2].

The *scaled* footrule distance is normalized to the range $[0,1]$ by dividing by an upper bound on the distance between the complete rankings: $F_s(r_1,r_2) = 2F(r_1,r_2)/|U|^2$. The scaled distance is useful for comparing the effectiveness of rankings across lists of different sizes [3].

In this paper, we will often need to assess the total distance between an aggregate ranking σ and a set R of k expert rankings $R = \{r_1 \ldots r_k\}$. When each of the experts produces a complete ranking, then the total distance is defined by: $F(\sigma,R) = \frac{1}{k}\sum_{j=1}^{k} F(\sigma,r_j)$. When the set R contains partial rankings, then define U as the union of all items in the lists in R, and if σ is a complete ranking on U then the *induced footrule distance* is computed via projections of σ onto the various lists in R: $F(\sigma,R) = \frac{1}{k}\sum_{j=1}^{k} F(\sigma_{|r_j}, r_j)$ [3].

Kendall Tau Distance. The *Kendall tau distance* between two ranked lists is defined as the number of pairwise disagreements in the relative rankings of items in the two lists. That is, $K(r_1,r_2) = |\{i,j\}$ s.t. $r_1(i) < r_1(j)$ and $r_2(i) > r_2(j)|$ [7]. (This is equivalent to the number of swaps that *bubble sort* must make to transform r_1 to r_2 [3].) As above, the Kendall tau distance may be scaled by the maximum possible disagreement, which is in this case $\binom{|U|}{2}$. Furthermore, an *induced Kendall tau* distance may be calculated in the same fashion as the induced footrule distance, which allows the computation of the total distance between an aggregate ranking σ and a set of k expert rankings R. In this case, let U be the union of all items in all the lists of R, ensure σ is a complete ranking on U, and compute $K_{ind}(\sigma,R) = \frac{1}{k}\sum_{j=1}^{k} K(\sigma_{|r_j}, r_j)$ [3].

3.2 Distance Measures with Similarity

These previous methods of measuring distances between ranked lists have a drawback for measuring distance between ranked lists of similar items when those lists are incomplete, as in the case of partial, top-k, or disjoint rankings. The drawback is that these methods ignore the rankings of items that do not appear in both lists. This makes sense when we do not have similarity infor-

mation – ignoring disjoint items may be the best solution. However, when we do have meaningful similarity information for ranked items, ignoring the effect of similar (but not identical) ranked items may cause us to inappropriately prefer one aggregation over another, or to be unable to distinguish between two aggregations. An example of these problems was shown in the introduction. To address these issues, we extend the standard distance measures to take similarity information into account.

Footrule Similarity Distance. We now define an extension of the Spearman footrule distance by including the effect of a similarity function defined on items in U. Note that we define this measure in terms of a generic similarity function. In practice, the choice of a particular similarity measure for domain is of great importance, and the development of similarity measures is an active area in the data mining and pattern recognition communities. Here, we take the similarity measure as given, and assume that the similarity scores returned are meaningful within the given domain.

DEFINITION 3.1. *The* footrule similarity distance *between two (possibly partial) ranked lists σ and r, given a similarity function $s(\cdot,\cdot)$, is defined as:*

$$F_{sim}(\sigma, r, s(\cdot,\cdot)) = \sum_{i \in \sigma_{\lambda|r}} \sum_{j \in r_{\lambda|\sigma}} s(i,j) |\sigma_{\lambda|r}(i) - r_{\lambda|\sigma}(j)|$$

That is, the footrule similarity distance is calculated on similarity projections of σ and r. The difference in ranks for items in these resulting lists is weighted by the strength of the similarity.

DEFINITION 3.2. *The* scaled footrule similarity distance *is defined as:*

$$F_{simScale}(\sigma, r, s(\cdot,\cdot)) = \frac{2F_{sim}(\sigma, r, s(\cdot,\cdot))}{|U|^2 \sum_{i \in \sigma_{\lambda|r}} \sum_{j \in r_{\lambda|\sigma}} s(i,j)}$$

In this case, U is union of the items in $\sigma_{\lambda|r}$ and $r_{\lambda|\sigma}$, and the term $\frac{|U|^2}{2} \sum_{i \in \sigma_{\lambda|r}} \sum_{j \in \sigma_{\lambda|\sigma}} s(i,j)$ is an upper bound on the maximum disagreement between the two lists.

DEFINITION 3.3. *The* induced footrule similarity distance *on a list σ and set of lists $r \in R$ is defined as:*

$$F(\sigma, R, s(\cdot,\cdot)) = \frac{1}{|R|} \sum_{r \in R} F_{sim}(\sigma, r, s(\cdot,\cdot));$$

A scaled induced footrule similarity distance *is similarly defined, by substituting $F_{simScale}$ for F_{sim}.*

Let's examine a few characteristics of these distances. First, when the uniqueness function $s_0(\cdot,\cdot)$ is used as the similarity function, $F_{sim}(\sigma, r, s_0(\cdot,\cdot)) = F(\sigma, r)$ and $F_{simScale}(\sigma, r, s_0(\cdot,\cdot)) = F_s(\sigma, r)$. That is, the original Spearman footrule distance can be viewed as special cases of these generalized distances incorporating the role of similarity. (However, the scaled footrule is not a special case of the scaled footrule similarity distance, due to differences in the denominator.)

Second, it is important to note that these measures are not *metric* in the formal sense. In particular, they may fail the identity requirement of formal metrics, as it is possible to have $F_{sim}(r, r, s(\cdot,\cdot)) \neq 0$ for particular choices of r, and $s(\cdot,\cdot)$. For example, consider the list:

r: A, B, C, A'

Here, the items **A** and **A'** have positive similarity under the given similarity measure. In this case, the footrule similarity distance from r to itself will be greater than zero, due to the disagreement in the rankings between the similar items **A** and **A'**. This is an important point: the similarity function is detecting an inconsistency in the expert's ranking. After all, if the similarity function is well chosen for a given domain, then similar items should have similar rankings. The footrule similarity distance decreases as the inconsistency is reduced.

Third, computing the footrule similarity distance is more expensive than the linear-time Spearman footrule distance. The footrule similarity measure between two lists r_1 and r_2 requires $O(|r_1||r_2|)$ evaluations of the similarity function. Thus, there is a premium on selecting similarity functions that are both informative and efficient.

Kendall Tau Similarity Distance. We now extend the Kendall tau distance to include item similarity. As before, we will require the specification of a similarity measure $s(\cdot,\cdot)$ on the items in U. Note that the Kendall tau distance examines the number of pairwise re-orderings between the two lists. Extending this measure raises the possible case that two items i and j in list r_1 may have many similar items $i'_1..i'_n$ and $j'_1..j'_m$ in r_2, and these similar items may be ordered in arbitrarily complex ways. Thus, we first define an *aggregate similarity position* function, $g(\cdot,\cdot,\cdot)$ and an *aggregate similarity list* r_g which will help resolve these complexities.

DEFINITION 3.4. *The aggregate similarity position of item i with respect to list r, under similarity function $s(\cdot,\cdot)$, is defined as*

$$g(i, r, s(\cdot,\cdot)) = \frac{\sum_{j \in r} s(i,j) r(j)}{\sum_{j \in r} s(i,j)}$$

An aggregate similarity list r_g is composed from lists r_1, r_2, and similarity function $s(\cdot,\cdot)$, such that for every element $i \in r_1$, $r_g(i) = g(i, r_2, s(\cdot,\cdot))$ when $g(i, r_2, s(\cdot,\cdot))$ is defined. When $g(i, r_2, s(\cdot,\cdot))$ is not defined, i is not in r_g. (Furthermore, note that rank values in r_g are not necessarily integers.) Such a list r_g is returned from the function $r_g(r_1, r_2, s(\cdot,\cdot))$.

DEFINITION 3.5. *The Kendall tau similarity distance between two (possibly partial) ranked lists r_1 and r_2 on items in U, given a pairwise similarity function $s(\cdot,\cdot)$ on items in U, is defined as*

$$K_{sim}(r_1, r_2, s(\cdot,\cdot)) = \frac{1}{2}\{K(r_1, r_g(r_1, r_2, s(\cdot,\cdot))) + K(r_2, r_g(r_2, r_1, s(\cdot,\cdot)))\}$$

Thus, $K_{sim}(r_1, r_2)$ is the average Kendall tau distance between r_1 and its aggregate similarity list drawn from r_2, and the Kendall tau distance between r_2 and its aggregate similarity list drawn from r_1.

As with the Footrule similarity distance, the original Kendall tau distance is recovered from the Kendall tau similarity distance by using the uniqueness function $s_0(\cdot,\cdot)$. Induced and scaled versions of the Kendall tau similarity distance are defined in similar terms. Note that scaled Kendall tau is not directly recoverable from the scaled Kendall tau similarity distance using $s_0(\cdot,\cdot)$.

DEFINITION 3.6. *The* induced *Kendall tau similarity distance is computed in the same way as the Kendall tau similarity distance, with $K_i(\cdot,\cdot)$ replacing $K(\cdot,\cdot)$.*

Similarly, the scaled *Kendall tau similarity distance is computed with $K(\cdot,\cdot)$ replacing $K_s(\cdot,\cdot)$.*

Unlike the Footrule similarity distance, the identity $K_{sim}(i, i, s(\cdot,\cdot)) = 0$ holds. However, both $K_{sim}(\cdot,\cdot, s(\cdot,\cdot))$ and $K(\cdot,\cdot)$ are non-metric for partial rankings, as either will compute zero distance between any list and the empty list. Finally, the evaluation of the Kendall tau similarity metric also requires $O(|r_1||r_2|)$ similarity computations, in addition to the two order $n \log n$ time evaluations of the Kendall tau distance.

3.3 Minimizing Distances Selecting a distance measure on ranked lists, our goal in rank aggregation is to find an aggregate list σ that minimizes the total distance from σ to every given list $r \in R$, as computed by the chosen distance measure.

Minimizing the Kendall tau distance results in what is called a *Kemeny optimal ranking*, which has well studied desirable properties, such as the Condorcet criterion which states that the top ranked item in aggregation should be the item that is preferred in a majority of pairwise comparisons [15]. Unfortunately, it has been shown that optimizing a Kemeny optimal ranking from as few as four lists is NP-hard [3]. In our setting of rank aggregation in the presence of similarity information, we see that minimizing the the Kendall-tau similarity distance is also NP-hard, as optimizing this distance requires optimizing the Kendall-tau distance on aggregate similarity lists.

4 Methods and Algorithms

Rank aggregation is a difficult problem even without the presence of similarity information, and there is no universally accepted best method. Moreover, rank aggregation among similar items is, to our knowledge, previously unstudied. Thus, we have developed a number of different methods to approach this problem, which draw from three main families of rank aggregation algorithms: positional methods, Markov chain methods, and median-rank methods. Our goal in this section is not to present a single algorithm for all situations, but rather to explore a diverse range of reasonable approaches to a challenging new problem.

4.1 Borda's Method. Borda's method is an intuitive method of rank aggregation first proposed for electing members of to the Academy of Science in Paris in 1770. Borda's method relies on the absolute position of items in the ranked lists, rather than their relative rankings, and is a classic representative of the class of *positional* ranking methods. Positional rank aggregation methods are generally computationally efficient, but no positional method can produce rankings guaranteed to satisfy the Condorcet criterion [15].

Borda's method assigns ranks to items based on a total Borda score, which is computed on each list by showing that the most preferred item in a universe of U items gets $|U|$ points, the next gets $|U| - 1$ points, and so on.

More formally, Borda's method on a set of complete rankings R is computed as follows. For each item i and list $r_k \in R$, let $B_{r_k}(i)$ equal the number of items j in r_k such that $r_k(j) > r_k(i)$. The total Borda score for the item i is given by $B_t(i) = \sum_{r \in R} B_r(i)$. Ranks are assigned by sorting scores B_t from highest to lowest, with the highest score getting the lowest rank.

When R includes partial rankings, one proposal is to assign any "leftover" score from a given list equally among all remaining unranked items in the list [12]. That is, for a list r_k, where $|r_k| = |U| - d$, compute the $B_{r_k}(i)$ as usual for all items $i \in r_k$, and assign $B_{r_k}(j) = \frac{(d+1)^2 - (d+1)}{2d}$ for all items $j \notin r_k$. The Borda scores B_t and rankings are assigned as above.

Borda's Method with Similarity (BMS). We extend Borda's method to include the application of similarity information. The basic idea is to create a weighted average of Borda scores based on the similarity between items. To this end, the BMS method aggregates a set of ranked lists $r \in R$ from a universe of item $i \in U$, by first computing the Borda scores $B_t(i)$ for all items in U, using either the method for complete or partial listings (above), as appropriate. Then, for each item i, compute a Borda-Similarity score, given a similarity function $s(\cdot, \cdot)$,

$$B_s(i) = \sum_{j \in U} \frac{s(i,j)B_t(j)}{\sum_{k \in U} s(i,k)}$$

and sort the items by B_s and assign ranks as with the original Borda method. Notice that when the uniqueness function $s_0(\cdot, \cdot)$ is used, the rankings given by BMS are identical to those given by Borda's method.

Borda's method computable in linear time (given sorted rankings); BMS adds a number of similarity evaluations that is quadratic in $|U|$. However, Borda's method does not always perform well with partial rankings, as the method is unable to propagate rankings. This issue is addressed with the introduction of Markov chain methods for rank aggregation.

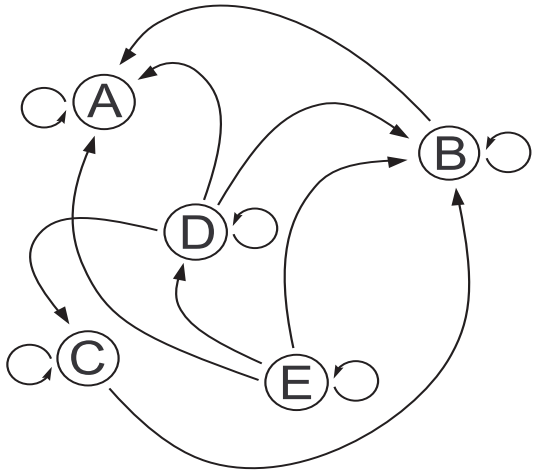

Figure 1: **Rank Aggregation with Markov Chains.** This Markov Chain represents the aggregation of lists **(B, C, D)**, **(A, D, E)**, and **(A, B, E)**. Exact probabilities of transitions are determined by the choice of mapping method. Self-transitions represent the probability of staying at the node during a step.

4.2 Markov Chain Methods The Markov Chain methods for ranked list aggregation represent the items in the various lists as nodes in a graph, with transitions probabilities from node to node defined by the relative rankings of the items in the various lists. The aggregate rankings of the lists are found by computing (or approximating) the stationary distribution on the Markov chain – that is, by determining which nodes would be visited most often in a random walk on the graph. We augment these methods to include the case of similar items by adding *epsilon transitions* between nodes, based on similarity between the corresponding items. We will formalize these notions in this section, starting with a brief review of basic characteristics of Markov chains, and moving on to the four algorithms. We will conclude with discussions of methods for efficiently computing (or approximating) the stationary distribution across the Markov chains.

Markov Chains. Markov chains are well studied models of stochastic processes. Here we review a few of their basic characteristics that are salient to the methods of rank aggregation.

A Markov chain M consists of a set of states $S = 1..n$ and a transition probability matrix P of dimension $n \times n$. Markov chains function in discrete time steps, and $Pr(S_t = i)$ is defined as the probability that a random walker on the chain will be at state i at time step t. The transition probabilities are defined by

$$P_{i,j} = Pr(S_t = j | S_{t-1} = i)$$

(Note that each row in P sums to 1.) Thus, Markov chains are *memoryless*: the probability of moving to any given state at the next time step depends only upon the current state, not on any previous states.

The *stationary distribution* on a given Markov chain is the set of probabilities: $Pr(S_\infty = i)$ for all $i \in S$. All finite, irreducible, ergodic Markov chains have a single stationary distribution, which may be computed by finding the principle eigenvector of the matrix P, or closely approximated taking higher powers of P to convergence or by simulating a random walk over M for a large number of steps. For large chains, simulating the random walk reduces computational cost considerably.

Markov Chain Rank Aggregation. When transition probabilities are defined by relative rankings, the stationary distribution on a Markov chain implies an ordering on the nodes in M (see Figure 1). This observation led Dwork et al. to propose a general algorithm for rank aggregation, composed of three steps [3]:

1. Map the set of ranked lists R to a single Markov chain M, with one node per item in U.

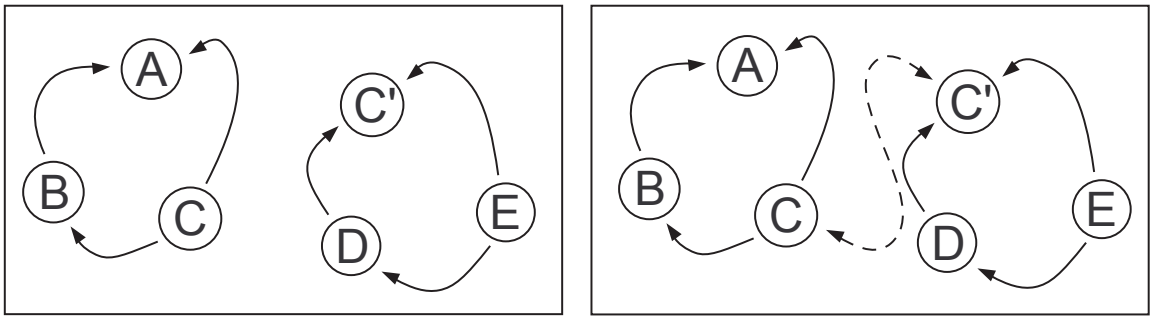

Figure 2: **Markov Chain Rank Aggregation with Similarity.** On the left, the Markov Chains are not able to merge the lists **A, B, C** and **C', D, E**, because the lists are completely disjoint. On the right, adding the dashed line of the similarity transition between **C** and **C'** enables a smooth ranking across all items. Note that epsilon transitions and self-transitions have been removed for clarity.

2. Compute (or approximate) the stationary distribution π on M.

3. Rank the items in U from $1..n$ based on π. That is, the node with the highest score in π is given rank 1, and so on down to the node with the lowest score in π which is given rank n.

The key to this method is to define the appropriate mapping from the set of ranked lists R to a Markov Chain M. Dwork et al. proposed, analyzed, and tested four mapping schemes, which they dubbed MC1 through MC4 [3]. Although none of these methods necessarily produces a Kemeny optimal ranking, these methods have proven effective in practice [3].

Markov Chain Rankings with Similarity. Although effective in many situations, the Markov Chain methods fail when the input lists are disjoint. (See Figure 2.) We address this issue with the addition of similarity transitions, which can link together otherwise isolated Markov Chain islands.

Similarity Transitions. Similarity transitions are defined from node to node based on the similarity measured between nodes. Thus, the ranking of an item will depend not only on those items it is ranked higher or lower than, but will also depend on the rankings of items that are similar.

A similarity transition from $S_k = i$, with respect to a given similarity function $s(\cdot, \cdot)$ is executed by choosing an item j from U randomly from a weighted distribution in which for all $j \in U$

$$Pr(i \rightarrow j) = \frac{s(i,j)}{\sum_{l \in U} s(i,l)}$$

Note that if the similarity function is the uniqueness function $s_0(\cdot, \cdot)$, then a similarity transition will always result in $S_{k+1} = S_k$.

Furthermore, note that we also include small, uniform probability *epsilon transitions* from every node to every other node. These eliminate the possibility of *sink nodes* in the Markov chain, and ensure a smooth, complete ranking on all the items in U. This smoothing technique appears in a number of Markov chain based ranking methods, including the Google PageRank algorithm [11].

Formally, an epsilon transition is executed at S_k by choosing an item j from U uniformly at random, and setting $S_{k+1} = j$.

The prior mappings MC1 through MC4 are generalized to consider similarity information, resulting in the following mappings MCS1 through MCS4. Note that each of these mappings include two parameters: ϵ determines the probability of an epsilon transition, while γ determines the probability of a similarity transition.

MCS1: At $S_k = i$, transition to S_{k+1} as follows. Create a multiset A of all nodes j such that there is a list $r \in R$ such that $r(j) \leq r(i)$, and choose a candidate node j by selecting uniformly at random from the nodes in A. Let $S_{k+1} = j$ if $i \neq j$. Otherwise, execute a similarity transition with probability γ. If no similarity transition is executed, then execute an epsilon transition with probability ϵ. If no epsilon transition is executed, let $S_{k+1} = i$.

This method pays a relatively small amount of attention to similarity information, and gives more weight to the relative rankings.

MCS2: At $S_k = i$, transition to S_{k+1} as follows. Choose a list r uniformly at random from the subset of the (possibly partial) lists $r \in R$ for which $r(i)$ is defined. Choose a node for S_{k+1} by choosing uniformly at random from the items j such that $r(j) \leq r(i)$. Let $S_{k+1} = j$ if $i \neq j$. Otherwise, execute a similarity transition with probability γ. If no similarity transition is executed, then execute an epsilon transition with probability ϵ. If no epsilon transition is executed, let $S_{k+1} = i$.

Dwork et al. show that **MC2** is a generalization of the geometric mean variant of Borda's method and argue that this method is most sensitive of the four to statistically significant minority opinions [3].

MCS3: At $S_k = i$, transition to S_{k+1} as follows. Choose a list r uniformly at random from the set of lists $r \in R$ such that $r(i)$ is defined. Choose an item j from r uniformly at random. If $r(j) < r(i)$, $S_{k+1} = j$, otherwise execute a similarity transition with probability γ. If no similarity transition is executed, then execute an epsilon transition with probability ϵ. If no epsilon transition is executed, let $S_{k+1} = i$.

MCS4: At $S_k = i$, transition to S_{k+1} as follows. Choose an item j uniformly at random from U. if $r(j) < r(i)$ for a majority of the lists that ranked both i and j, $S_{k+1} = j$, otherwise execute a similarity transition with probability γ. If no similarity transition is executed, then execute an epsilon transition with probability ϵ. If no epsilon transition is executed, let $S_{k+1} = i$.

Dwork et al. show that this method is a generalization of Copeland's method of ranking items based on the number of pairwise majority contests won [3].

4.3 Median Rank Aggregation.

The third method of rank aggregation we explore is that of *median rank aggregation*, which is to aggregate a set of complete rankings by using the median rank for each item. This method can produce footrule optimal aggregations, which are within a constant bound of Kemeny optimal. The rankings it produces satisfy the extended Condorcet criterion, and it may be computed efficiently, especially in the case where only the top k aggregate rankings are required [4].

The heart of the MEDRANK algorithm is as follows [4]. To compute an aggregate ranking on items $i \in U$ from a set of complete ranked lists $r \in R$, begin by initializing a set of scores $M(i) = 0$ for all $i \in U$. Let the function $c(i, n)$ return the number of of lists in R for which $r(i) = n$. Starting at $n = 1$, compute $M(i) = M(i) + c(i, n)$ for all $i \in U$, incrementing n at each step. The first item i with a score $M(i) > \theta$ gets rank 1, the second such item gets rank 2, and so on, with ties being arbitrarily broken. With standard MEDRANK, $\theta = \frac{|R|}{2}$; thus, an item must appear in at least half the lists before getting an aggregate rank.

The MEDRANK algorithm thus creates an aggregate ranking based on the median ranks of items in the set of lists R, and does so in linear time.[1] When only the top k rankings are desired, the algorithm will terminate early, and not every item in every list will be explicitly examined, giving sub-linear evaluation time. Furthermore, when the aggregate rankings are all distinct (no ties), MEDRANK produces a footrule optimal ranking, which is within a constant bound of the Kemeny optimal ranking [15].

When using MEDRANK of partial rankings, some of these theoretical guarantees are lost. To modify MEDRANK for use on partial rankings, we can set θ to a value below $\frac{|R|}{2}$. Similarly, Fagin et al. also note the possibility of setting $\theta > \frac{|R|}{2}$, and expect improved results [4] – although increasing θ may not necessarily increase the quality of aggregating very noisy rankings.

Median/Similarity Rank Aggregation. We propose a modification of MEDRANK, which we call SIMMEDRANK, to include the effect of similarity in rank aggregation. The basic idea is to assign the highest aggregate rank to the first item that achieves a certain total similarity score γ in each of θ lists $r \in R$.

The method SIMMEDRANK works as follows. For each list $r \in R$, and each item $i \in U$, set the total similarity score $t_s(r, i) = 0$. Starting at $n = 1$, let $t_s(r, i) = t_s(r_i) + s(i, r^n)$ for all i and r, using a similarity function $s(\cdot, \cdot)$. (Recall that r^n refers to the *item* in r at rank n.) Let the function $M_s(i, n)$ return the number of lists r such that $t_s(r, i) \geq \gamma$ at step n. The item i for which $M_s(i, n) > \theta$ with the smallest value of n is given aggregate rank 1, and so on. (Ties may be broken first by comparing $\sum_{r \in R} t_s(r, i)$ at step n, with further ties being broken arbitrarily.)

When the uniqueness function $s_0(\cdot, \cdot)$ is used, setting $\gamma = 1$ reduces to the original MEDRANK method. For other similarity measures and data sets, γ must be chosen. (In absence of other information, $max(s(i, i))$ over $i \in U$ is a reasonable starting value.) As before, SIMMEDRANK may be used with partial lists by reducing the threshold value θ, although the standard version uses $\theta = \frac{|R|}{2}$. When used to produce a complete aggregate ranking over U, SIMMEDRANK will require an additional quadratic number of similarity evaluations. However, this number will be reduced if only the top k results from the aggregate ranking are needed.

[1] Fagin et al. show efficient algorithms for computing MEDRANK on large databases using very few random accesses [4].

5 Experiments

In this section, we report the results on both real world and synthetic data. We find that the addition of similarity information to rank aggregation is a significant benefit to noisy, incomplete rankings. Furthermore, we find empirical support for the use of our distance measures with similarity.

The following parameters were used in our various aggregation methods, all of which were set by testing on small, independent tuning sets. The value of λ in λ-similarity projections was zero. For the Markov Chain methods, $\epsilon = .01$, and $\gamma = 1$, and the stationary distribution was approximated by the power method. For the median rank methods, $\theta = \frac{|R|}{2}$, and for SIMMEDRANK, $\gamma = 1$.

5.1 Synthetic Data We can view the goal of rank aggregation in two ways. The goal may be seen as trying to find a complete ranking σ on U that minimizes the induced distance between σ and the input lists $r \in R$. Or, we can imagine that there is some true ranking r_t on U, and that each given list $r \in R$ is a distortion of r_t, possibly with missing items and noisy rankings. In this case, the goal is to minimize the distance between σ and r_t – that is, to recover the true ranking from the set of noisy rankings. Many evaluations of rank aggregation methods have focused on the first version of the problem [3], in part because it is difficult to know the true ranking r_t for real-world problems. However, it is possible to create a test set of rankings R by randomly distorting a given input ranking r_t. Synthetic data allows reasonable comparison between the output σ to the true ranking r_t.

Synthetic Data Sets. Because each of the rank aggregation algorithms may react differently to noisy rankings, partial rankings, and top-k rankings, we created a set of test data for each situation.

For each set, we began with an initial ranked list r_t, composed of 100 items, which were grouped into twenty families $a..t$ of five items each. (These numbers were chosen arbitrarily.) We defined a similarity function $s_f(\cdot,\cdot)$ such that $s_f(i,i) = 1$ for all $i \in U$, $0 < s_f(i,j) < 1$ for all (i,j) such that i and j are in the same family and $i \neq j$, and $s(i,j) = 0$ for all i and j in different families. The list r_t was formed by ranking the items in order by family: $(a_1..a_5, b_1..b_5, ..., t_1..t_5)$. Thus, r_t was constructed such that similar items have similar ranks. We then created test data as follows.

Noise. For each test set of complete rankings, we created a set of ten complete lists, each containing noisy versions of r_t. Noise was introduced by randomly swapping pairs of rankings independently in each list. We created test sets at five levels of noise: 10, 25, 50, 75, and 100 random swaps per list.

Partial. For each test set of partial rankings, we created a seed set of ten noisy, complete rankings, as above. We then pruned each ranking by removing each item i with probability $1 - p$. We used three levels of noise (0, 10, 25), and five levels of p (0.1, 0.2, 0.3, 0.4, 0.5), starting with new seed lists each time.

Top-k. For each test of top-k rankings, we first created a set of ten noisy, complete rankings as above. We then selected the top k of these items from each list. We created lists at noise levels of 0, 10, and 25, and with k at levels of 10, 25, 50, and 100, starting with new seed lists each time.

We ran each rank aggregation method on each data set, and report the results in Table 1, Table 2, and Table 3. We report results for the induced scaled Kendall tau similarity distance, noting that these results were consistent with the induced scaled Footrule similarity distance and with non-scaled versions of each measure. Furthermore, every improvement in both measures between σ and R corresponded with an improvement between σ and the ground truth r_t.

Synthetic Results. Across these tests, each method of rank aggregation with similarity was consistently superior to the corresponding method without similarity. These results show the potential for improvement when a meaningful similarity measure is defined on a set of ranked items.

The **Noise** results in Table 1 show that using rank aggregation with similarity gives improvement for all methods tested (except for MEDRANK with low noise levels), and that this benefit increases as noise increases.

The **Partial** results in Table 2 show increasing benefit to using rank aggregation with similarity as the coverage of the partial lists increases. We conjecture that at very low levels of coverage, such as the $p = 0.1$ level, there may be very few items with non-zero similarity in the lists. Note also that the Markov Chain methods out-perform Borda's method and MEDRANK, which matches the intuition that the Markov Chain methods are well suited to partial lists.

The **Top-k** results in Table 3 also show increasing benefit as k increases. Here, the Markov Chain methods also out perform the other two methods. These results were consistent at other noise levels.

Evaluating the Distance Measures. At this point, it is appropriate to question the effectiveness of our evaluation measures. How do we know that the Kendall tau similarity distance and the Footrule similarity distance are meaningful success measures?

Table 1: **Aggregating Noisy Rankings.** Results are given for aggregating complete lists (of 100 items) at varying levels of noise. Results reported are for the scaled Kendall-Tau similarity distance, induced between aggregate list σ and a set of ten input lists $r \in R$, averaged over 50 independent trials. (Recall that the ideal value is zero.) Results are in format: mean (standard deviation).

Noise Level:	10	25	50	75	100
BORDA	.123 (.016)	.224 (.017)	.339 (.021)	.395 (.017)	.418 (.017)
BMS	.106 (.014)	.194 (.015)	.299 (.019)	.353 (.019)	.386 (.019)
MC3	.135 (.017)	.230 (.016)	.340 (.021)	.395 (.017)	.418 (.017)
MCS3	.120 (.014)	.203 (.014)	.306 (.019)	.361 (.019)	.386 (.019)
MC4	.104 (.016)	.216 (.019)	.339 (.020)	.396 (.017)	.419 (.017)
MCS4	.104 (.017)	.208 (.017)	.310 (.019)	.364 (.019)	.389 (.019)
MEDRANK	.103 (.016)	.212 (.018)	.343 (.020)	.401 (.018)	.426 (.017)
SIMMEDRANK	.107 (.017)	.228 (.024)	.345 (.022)	.391 (.021)	.413 (.020)
MC1	.147 (.020)	.237 (.016)	.342 (.021)	.396 (.017)	.419 (.017)
MCS1	.146 (.019)	.235 (.016)	.340 (.021)	.395 (.018)	.417 (.017)
MC2	.155 (.022)	.256 (.016)	.359 (.022)	.408 (.018)	.429 (.017)
MCS2	.160 (.023)	.256 (.021)	.356 (.022)	.404 (.019)	.425 (.018)

Table 2: **Aggregating Partial Rankings.** Results are given for aggregating partial lists of various size at noise level 25. (Results consistent with tests at other noise levels.)

p:	0.1	0.2	0.3	0.4	0.5
MC4	.041 (.011)	.088 (.015)	.125 (.017)	.152 (.016)	.164 (.018)
MCS4	.034 (.009)	.071 (.013)	.100 (.014)	.100 (.014)	.137 (.022)
MC3	.041 (.011)	.090 (.016)	.128 (.018)	.157 (.017)	.171 (.017)
MCS3	.035 (.010)	.076 (.010)	.106 (.015)	.131 (.016)	.142 (.015)
MC2	.043 (.012)	.094 (.017)	.136 (.019)	.167 (.018)	.187 (.018)
MCS2	.041 (.012)	.090 (.015)	.128 (.019)	.161 (.018)	.181 (.018)
MC1	.042 (.012)	.094 (.017)	.136 (.019)	.166 (.017)	.184 (.017)
MCS1	.040 (.012)	.094 (.015)	.129 (.019)	.160 (.017)	.178 (.017)
BORDA	.055 (.013)	.124 (.019)	.177 (.120)	.177 (.019)	.231 (.018)
BMS	.059 (.015)	.123 (.022)	.154 (.023)	.154 (.023)	.183 (.022)
MEDRANK	.057 (.013)	.126 (.020)	.182 (.020)	.182 (.020)	.245 (.019)
SIMMEDRANK	.063 (.015)	.133 (.020)	.159 (.022)	.159 (.022)	.151 (.018)

Table 3: **Aggregating Top-k Rankings.** Results are given for aggregating sets of top-k lists, where the maximum list size is 100, at noise level 25. (Results consistent with tests at other noise levels.)

k:	10	25	50	100
MC4	.036 (.013)	.076 (.013)	.140 (.016)	.216 (.019)
MCS4	.031 (.013)	.068 (.012)	.123 (.014)	.208 (.017)
MC3	.037 (.013)	.084 (.014)	.152 (.015)	.230 (.016)
MCS3	.036 (.014)	.078 (.014)	.135 (.014)	.203 (.015)
MC2	.041 (.014)	.094 (.017)	.167 (.017)	.256 (.020)
MCS2	.039 (.014)	.089 (.017)	.162 (.018)	.256 (.021)
MC1	.042 (.014)	.094 (.016)	.165 (.016)	.237 (.016)
MCS1	.039 (.014)	.090 (.017)	.161 (.016)	.235 (.016)
MEDRANK	.081 (.016)	.134 (.019)	.196 (.017)	.212 (.019)
SIMMEDRANK	.075 (.013)	.119 (.014)	.170 (.013)	.228 (.025)
BORDA	.086 (.016)	.143 (.019)	.205 (.015)	.224 (.017)
BMS	.079 (.021)	.132 (.020)	.189 (.016)	.194 (.015)

Table 4: **Evaluating Distance Measures.** Results on all distance measures for MC3 and MC3S, on partial lists, noise 25, p 0.5. Distance measured between aggregate list σ and input set of lists $r \in R$, and between σ and ground truth ranking r_t. Format is: mean (standard deviation). Results show that the similarity versions of distance measures are consistent in preferring similarity aggregation over non-similarity aggregation on all tests, while non-similarity distance measures are inconsistent. This pattern is repeated in many results across our tests.

	KENDALL-TAU	FOOTRULE	KENDALL-TAU SIM.	FOOTRULE SIM.
MC3 (LISTS)	.059 (.007)	.244 (.012)	.171 (.017)	.289 (.010)
MCS3 (LISTS)	.065 (.007)	.256 (.011)	.142 (.015)	.277 (.011)
MC3 (TRUTH)	.223 (.030)	.308 (.039)	.184 (.030)	.309 (.039)
MCS3 (TRUTH)	.146 (.015)	.212 (.033)	.141 (.024)	.213 (.033)

One way of confirming their usefulness empirically is with the results reported in Table 4. We compared the Kendall-tau and Footrule Distances (non-similarity versions) between the aggregate list σ and the input lists $r \in R$, and also between σ and the ground truth r_t. We would expect that if one method shows improvement on $K(\sigma, R)$, it would also show improvement on $K(\sigma, r_t)$. This was often not the case with the Kendall tau or Spearman Footrule in their original forms.

In our tests we have found that an improvement in $K_{sim}(\sigma, R)$ does correspond with an improvement in $K_{sim}(\sigma, r_t)$, and similarly so for the Footrule similarity distance. That is, reducing the similarity distance between an aggregate list and the set of input lists corresponds to a reduction in the distance between the aggregate list and the ground truth. This empirical test serves as an important sanity check for our evaluation measures – reducing the Kendall tau similarity distance or the footrule similarity distance creates an aggregation closer to ground truth.

5.2 Real World Data. Finally, we experiment with data drawn from a real-world application: keywords expansion for sponsored search. In sponsored search, a service provided by many Internet search engines such as Yahoo!, an advertiser places bids on a set of keywords. When a user searches for that keyword, the search engine may show the advertiser's listing in a premium area, and the advertiser will pay the bid price to the search engine if the user clicks on the listing.

It is advantageous to both advertisers and search engines to make sure that the advertiser is bidding on as large a list of relevant keywords as possible – this reduces unused inventory and optimizes both market efficiency and user traffic. The goal of *keywords expansion* is to automatically suggest additional relevant keywords to a given advertiser. There are many methods for generating such suggestions, including the use of taxonomies, mining the text of the advertiser's website, and analyzing aggregate user behavior and keyword use. However, each of these method has incomplete coverage of the total keyword space and is subject to noise. The various methods of generating suggestions use different scoring mechanisms, on different scales, and perhaps with different levels of confidence. Finally, experience has shown that these lists of suggested keywords have little exact overlap, although many of the keywords share high levels of string similarity. For example, the keywords "Honda Civic" and "red Honda Civic" may appear as distinct items in different lists of suggested keywords. This is a perfect application for rank aggregation with similarity information, which can meaningfully join ranked lists of non-overlapping, similar items – even on partial lists in the presence of moderate noise.

We tested our methods on ten data sets, each with three ranked lists of keywords generated by distinct methods of keyword suggestion, some of which were quite noisy. Furthermore, there was little overlap among the lists in each set. The similarity measure used on the keywords was a simple n-gram string matching kernel [9], with $n = 2$. Other similarity measures, such as those based on edit distance, could also have been used.

Results. The results in Table 5 show that in this experiment, rank aggregation with similarity consistently out performed non-similarity aggregation methods, as measured by both the scaled footrule similarity distance, with SIMMEDRANK and MC4 showing the best overall results. Similar improvement was measured with the scaled Kendall tau similarity distance, although BMS performed relatively better and MC4 fared relatively worse when evaluated with this measure. This experiment serves as a real world example of noisy rankings, with a mixture of characteristics of partial and top-k lists, which benefits from the addition of similarity information to rank aggregation.

6 Discussion

The use of similarity information in rank aggregation has a wide range of potential applications. It could be used in spam reduction [3], importance ranking of

Table 5: **Aggregating Keyword Rankings.** Results for rank aggregation on keywords expansion data for scaled Footrule similarity distance. Results were consistent with those of the scaled Kendall similarity distance, except that BMS had relatively better performance on that measure, and MCS4 had relatively worse performance.

SET ID:	0	1	2	3	4	5	6	7	8	9
MEDRANK	0.237	0.237	0.234	0.230	0.248	0.211	0.242	0.251	0.225	0.242
SIMMEDRANK	0.189	0.194	0.200	0.186	0.206	0.174	0.174	0.195	0.191	0.237
MC4	0.242	0.257	0.255	0.245	0.268	0.208	0.266	0.256	0.242	0.242
MCS4	0.197	0.209	0.198	0.189	0.213	0.176	0.192	0.196	0.208	0.186
MC3	0.242	0.256	0.254	0.240	0.268	0.207	0.267	0.257	0.242	0.242
MCS3	0.209	0.227	0.218	0.203	0.226	0.181	0.212	0.201	0.216	0.203
MC2	0.240	0.253	0.253	0.246	0.271	0.207	0.279	0.260	0.242	0.242
MCS2	0.221	0.242	0.239	0.219	0.253	0.188	0.245	0.226	0.223	0.226
MC1	0.241	0.253	0.253	0.248	0.269	0.208	0.283	0.264	0.241	0.241
MCS1	0.224	0.246	0.240	0.218	0.255	0.191	0.256	0.238	0.223	0.224
BORDA	0.242	0.258	0.253	0.242	0.268	0.206	0.262	0.250	0.243	0.243
BMS	0.228	0.233	0.256	0.239	0.259	0.188	0.203	0.198	0.210	0.274

web pages [11], or to improve the quality of results in meta-search [3]. Indeed, Ravi Kumar has noted the possibility of defining a similarity measure that rewards *dis*-similarity, and using these methods to improve *diversity* within the top-k results in meta-search [8]. It is also possible to imagine using rank aggregation with similarity in the field of social choice, and designing elections in which party affiliation is used as a similarity measure between candidates.

In the fields of data mining and machine learning, rank aggregation with similarity has its strongest use as a method of unsupervised learning, in which items are implicitly clustered as an effective aggregate ranking is found. Future work will lie in further exploring ways in which these goals can be simultaneously achieved.

7 Acknowledgments

This work was performed in the Data Mining and Research group of Yahoo!, Inc., under the direction of Pavel Berkhin, Rajesh Parekh, and Scott J. Gaffney. Thanks are given to Anselm Blumer, Carla E. Brodley, and Ravi Kumar for their comments on this work.

References

[1] M.-J. Condorcet. *Essai sur l'application de l'analyse a la probabilite des decisions rendeus a la pluralite des voix*, 1785.

[2] P. Diaconis and R. L. Graham. *Spearman's Footrule as a Measure of Disarray* Journal of the Royal Statistical Society. Series B (Methodological) (1977), 39:2, pp. 262–268

[3] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. *Rank aggregation methods for the web*, WWW '01: Proceedings of the 10th international conference on World Wide Web, (2001), pp. 613-622.

[4] R. Fagin, R. Kumar and D. Sivakumar. *Efficient similarity search and classification via rank aggregation*, SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data (2003), pp. 301–312.

[5] R. Fagin, R. Kumar, and D. Sivakumar. *Comparing top-k lists.* In Proceedings of the 2003 ACM-SIAM Symposium on Discrete Algorithms (2003),

[6] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar and E. Vee. *Comparing and aggregating rankings with ties*, PODS '04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems (2004), pp. 47–58.

[7] M. G. Kendall. *A New Measure of Rank Correlation*, Biometrika (1938), 30:1/2, pp. 81–93.

[8] R. Kumar. Personal communication, May, 2006.

[9] C. Leslie, E. Eskin, and W. Noble. *The spectrum kernel: a string kernel for svm protein classification*, In Proceedings of the Pacific Symposium on Biocomputing (2002), pages 564–575.

[10] S. Nene and S. Nayar. *A Simple Algorithm for Nearest Neighbor Search in High Dimensions*, IEEE Transactions on Pattern Analysis and Machine Intelligence (1997), 19:9, pp. 989–1003.

[11] L. Page, S. Brin, R. Motwani, T. Winograd. *The PageRank citation ranking: Bringing order to the Web*, Technical report (1998), Stanford University, Stanford CA.

[12] D. G. Sarri. *The mathematics of voting: democratic symmetry.* Economist. p. 83, March 4, 2000.

[13] B. Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press (2001).

[14] J. Shawe-Taylor and N. Cristianini *Kernel Methods for Pattern Analysis* Cambridge University Press (2004).

[15] H. P. Young and A. Levenglick. *A Consistent Extension of Condorcet's Election Principle*, SIAM Journal on Applied Mathematics (1978), 35:2, pp. 285–300.