A Flexible Generative Model for Preference Aggregation

Maksims N. Volkovs
University of Toronto
40 St. George Street
Toronto, ON M5S 2E4
mvolkovs@cs.toronto.edu

Richard S. Zemel
University of Toronto
40 St. George Street
Toronto, ON M5S 2E4
zemel@cs.toronto.edu

ABSTRACT

Many areas of study, such as information retrieval, collaborative filtering, and social choice face the preference aggregation problem, in which multiple preferences over objects must be combined into a consensus ranking. Preferences over items can be expressed in a variety of forms, which makes the aggregation problem difficult. In this work we formulate a flexible probabilistic model over pairwise comparisons that can accommodate all these forms. Inference in the model is very fast, making it applicable to problems with hundreds of thousands of preferences. Experiments on benchmark datasets demonstrate superior performance to existing methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Experimentation, Theory

Keywords

Preference Aggregation, Meta Search, Collaborative Filtering

1. INTRODUCTION

Preference aggregation is the problem of combining multiple preferences over objects into a single consensus ranking. This problem is crucially important in many applications, such as information retrieval, collaborative filtering and social choice. Across various domains, the preferences over objects are expressed in several different ways, ranging from full and partial rankings to arbitrary comparisons. For instance, in meta-search an issued query is sent to several search engines and the (often partial) rankings returned by them are aggregated to generate more comprehensive ranking results. On the other hand, in online gaming the goal is typically to estimate the rank/skill of the players that participate in 1-on-1 games as well as tournaments. The resulting evidence of players' skill thus comes in the form of pairwise comparisons as well as partial tournament rankings, with many observations of the form "a beat b" and "b beat a".

Given the underlying correspondence between ranking and permutation, considerable work in machine learning has exploited probabilistic models on permutations, many of which originate in statistics and psychology. Mallows [18] and Plackett-Luce [22, 17] are particularly popular models, each with many extensions [9, 23, 16]. However, research has largely concentrated on learning a consensus ranking based on a set of observed full, or partial rankings. These models are thus inadequate for problems where preferences are expressed in other forms, and where inconsistencies exist in the observed preferences, such as "a beat b", "b beat c", and "c beat a".

In this paper we address this problem by developing a flexible probabilistic model over pairwise comparisons. Pairwise comparisons are the building blocks of almost all forms of evidence about preference and subsume the most general models of evidence proposed in literature. Our model can thus be applied to a wide spectrum of preference aggregation problems and does not impose any restrictions on the type of evidence. The score-based approach that we adopt allows for rapid learning and inference, which makes the model applicable to large-scale problems with hundreds of thousands of preferences. Experiments on a meta-search task with Microsoft's LETOR4.0 [14] data sets show that our model outperforms existing state-of-the-art methods designed specifically for this task.

2. FRAMEWORK

We assume a set of M items $X = \{x_1, ..., x_M\}$ and a set of N agents. Each agent n generates a list of preferences for items in X. The preferences can be in the form of full or partial rankings, ratings, relative item comparisons, or combinations of these. All of these forms can be converted to a set of *partial pairwise preferences*, which in most cases will be neither complete nor consistent. We use $\{x_i \succ x_j\}$ to denote the preference of x_i over x_j. We allow the same pairwise preferences to occur multiple times, and use the pairwise count matrix $C_n(i,j) : M \times M$ to count the number of times preference $\{x_i \succ x_j\}$ is produced by the agent n, with $C_n(i,j) = 0$ if $\{x_i \succ x_j\}$ is not expressed by n. The most straightforward way to convert rankings into pairwise preferences is through binary comparisons. Given two rankings r_{ni} and r_{nj} assigned by n to x_i and x_j we set $C_n(i,j) = I[r_{ni} < r_{nj}]$ where I is an indicator function, similarly $C_n(j,i) = I[r_{ni} > r_{nj}]$. This representation, however, completely ignores the strength of preference expressed by the magnitude of the rankings. For example, the partial ranking $\{1, 200, 300\}$ will have the same count matrix as the ranking $\{1, 2, 3\}$, but the first ranking expresses significantly more confidence about the ordering of the items than the second one. To account for this we instead use $C_n(i, j) = (r_{nj} - r_{ni})I[r_{ni} < r_{nj}]$ and $C_n(j, i) = (r_{ni} - r_{nj})I[r_{ni} > r_{nj}]$. In this form we assume that ranking $\{r_{ni} = 1, r_{nj} = 200\}$ is equivalent to observing the pairwise preference $\{x_i \succ x_j\}$ 199 times, whereas ranking $\{r_{ni} = 1, r_{nj} = 2\}$ is equivalent to observing $\{x_i \succ x_j\}$ only once. This method of accounting for preference strength is not new and the reader can refer to [8, 11] for more extensive treatment of this and other approaches for converting rankings to pairwise matrices.

A ranking of items in X can be represented as a permutation of X. A permutation π is a bijection $\pi : X \to \{1, ..., M\}$ mapping each item x_i to its rank $\pi(i)$, and $x_i = \pi^{-1}(i)$. Given the observed (partial) preference instance consisting of count matrices $\{C_1, ..., C_N\}$ the goal is to come up with a single ranking π of items in X that maximally satisfies this instance.

Most preference aggregation problems fit this framework. For instance in meta-search X is the set of documents retrieved for a given query. Each search engine n generates either partial or complete ranking of the documents in X. As before we can let $C_n(i, j) = (r_{nj} - r_{ni})I[r_{ni} < r_{nj}]$ if documents x_i and x_j are both ranked by the search engine n and set $C_n(i, j) = 0$ otherwise. In collaborative filtering X is the set of movies/songs/books etc., and an instance of the rank aggregation problem aims to infer the consensus ranking of movies given the (partial) ratings of N users [8, 9]. The pairwise approach provides a natural way to model this problem. We can define $C_n(i, j) = (l_{ni} - l_{nj})I[l_{ni} > l_{nj}]$ where l_{ni} and l_{nj} are the ratings assigned to movies x_i and x_j by user n. If n did not rate either x_i or x_j we set $C_n(i, j)$ to 0.

Finally, we note that in some settings, there are multiple preference aggregation problems. In meta-search for example, the same set of N search engines are the agents for multiple queries, returning a set of partial rankings of the M documents for each query. We use $S^{(\ell)}$ and $\mathbf{C}^{(\ell)} = \{C_1^{(\ell)}, ..., C_N^{(\ell)}\}$ to denote the scores and pairwise counts for each query ℓ.

3. RELATED WORK

Relevant previous work in this area can be divided into two categories: permutation based and score based. In this section we briefly describe both types of models.

3.1 Permutation Based Models

Permutation based models work directly in the permutation space. The most common and well explored such model is the Mallows [18] model. Mallows defines a distribution over permutations and is typically parametrized by a central permutation σ and a dispersion parameter $\phi \in (0, 1]$; the probability of a permutation π is given by:

$$P(\pi|\phi, \sigma) = \frac{1}{Z(\phi, \sigma)} \phi^{-d(\pi,\sigma)} \quad (1)$$

where $d(\pi, \sigma)$ is a distance between π and σ. For rank aggregation problems inference in this model amounts to finding the permutation σ that maximizes the likelihood of the observed rankings. For some distance metrics, such as Kendall's tau (the difference between the proportion of item pairs in the correct versus incorrect order w.r.t. σ), the partition function $Z(\phi, \sigma)$ can be found exactly. However, finding the central permutation σ that maximizes the likelihood is typically very difficult and in many cases is intractable [21].

Recent work extends the Mallows model to define distributions over partial rankings [16]. Under partial rankings the partition function can no longer be computed exactly, so these authors introduced a new sampling approach to estimate it. When M is large, however, this sampling approach is typically very slow, which makes the model impractical for many large scale online problems such as meta-search where aggregation has to be done very quickly. Furthermore, both the proposed pairwise model and the sampling approach rely on the assumption that all pairwise preferences are consistent, which is often violated in real-world preference aggregation problems.

Several other generalizations of the Mallows model such as the CPS model [23], the Aggregation model [12] and the Cranking model [13] have recently been explored. Due to space limitations we only discuss the CPS model here. CPS defines a sequential generative process, similar to the Plackett-Luce model described below, which draws the items without replacement to form a permutation; the probability of a given permutation π is:

$$P(\pi|\sigma, \phi) = \prod_{i=1}^{M} \frac{\exp(-\phi \sum_{r \in \Omega_{\pi_{1:i}}} d(r, \sigma))}{Z(i, \pi)} \quad (2)$$

where $\Omega_{\pi_{1:i}}$ is a set of permutations where the first i positions are fixed to π; $Z(i, \pi)$'s are the normalizing constants that ensure that $\sum_{\pi} P(\pi|\sigma, \phi) = 1$. For several distance metrics such as Spearman's rank correlation and footrule as well as Kendall's tau, the summation $\sum_{r \in \Omega_{\pi_{1:i}}} d(r, \sigma)$ over $(M - i)!$ elements, can be found in $O(M^2)$, allowing the normalizing constants $Z(i, \pi)$ to be computed in polynomial time. However, during inference one must still consider nearly all of the $M!$ possible permutations to find an optimal σ. A greedy approximation avoids this search, which reduces the complexity to $O(M^2)$, but provides no guarantee with respect to the optimal solution.

In general, due to the extremely large search space (typically $M!$ for M items) and the discontinuity of functions over permutations, exact inference in permutation based models is often intractable. Thus one must resort to approximate inference methods, such as sampling or greedy approaches, often without guarantees on how close the approximate solution will be to the target optimal one. As the number of items grows, the cost of finding a good approximation increases significantly, which makes the majority of these models impractical for real world applications where data collections are extremely large. The score based approach described in the next section avoids this problem by working with real valued scores instead.

3.2 Score Based Models

In score based approaches the goal is to learn a set of real valued scores (one per item) $S = \{s_1, ..., s_M\}$ which are then used to sort the items. Working with scores avoids the discontinuity problems of the permutation space.

Early score based methods for rank aggregation in meta search are heuristic based. For example, BordaCount [1] and median rank aggregation [7] derive the item scores by averaging ranks across the agents or counting the number of pairwise wins. In statistics a very popular pairwise score model is the Bradley-Terry [3] model:

$$P(C_n|S) = \prod_{i \neq j} \left(\frac{\exp(s_i)}{\exp(s_i) + \exp(s_j)} \right)^{C_n(i,j)} \quad (3)$$

where $\frac{\exp(s_i)}{\exp(s_i)+\exp(s_j)}$ can be interpreted as the probability that item x_i beats x_j in the pairwise contest. In logistic form the Bradley-Terry model is very similar to another popular pairwise model, the Thurstone model [25]. Extensions of these models include the Elo Chess rating system [6], adopted by the World Chess Federation FIDE in 1970, and Microsoft's TrueSkill [5] rating system for player matching in online games, used extensively in Halo and other games. Furthermore, the popular learning-to-rank model RankNet [4] is also based on this approach.

The key assumption behind the Bradley-Terry model is that the pairwise probabilities are completely independent of the items not included in the pair. A problem that arises from this assumption is that if a given item x_i has won all pairwise contests, the likelihood becomes larger as s_i becomes larger. It follows that a maximum likelihood estimate for s_i is ∞ [20]. As a consequence the model will always produce a tie amongst all undefeated items. Often this is an unsatisfactory solution because the contests that the undefeated items participated in, and their opponents' strengths, could be significantly different.

To avoid some of these drawbacks, the Bradley-Terry model was generalized by Plackett and Luce [22, 17] to a model for permutations:

$$P(\pi|s) = \prod_{i=1}^{M} \frac{\exp(s_{\pi^{-1}(i)})}{\sum_{j=i}^{M} \exp(s_{\pi^{-1}(j)})} \quad (4)$$

The generative process behind the Plackett-Luce model assumes that items are selected sequentially without replacement. Initially item $\pi^{-1}(1)$ is selected from the set of M items and placed first, then item $\pi^{-1}(2)$ is selected from the remaining $M-1$ items and placed second and so on until all M items are placed. Note that here inference can be done quickly by doing simple gradient descent on scores, which is a clear advantage over most permutation based models. The Plackett-Luce generalization relaxes the independence assumption of the Bradley-Terry model but this model is only applicable to consistent full or partial rankings (or consistent pairwise preferences) which significantly limits its application.

Recently several score based approaches have been developed to model the joint pairwise matrix [8, 11]. In these methods the preferences expressed by each of the N agents are combined into a single preference matrix $Y : M \times M$, which is then factorized by a low rank factorization such as:

$$Y = S\mathbf{e}^T - \mathbf{e}S^T$$

The resulting scores S are then used to rank the items. The main drawback of this approach is that by combining all preferences into a single Y the individual user information is lost. Consequently outlier agents with preferences substantially deviating from the consensus can significantly influence both Y and the resulting scores.

A supervised score based rank aggregation approach was also recently introduced [15]. In this model the ground truth

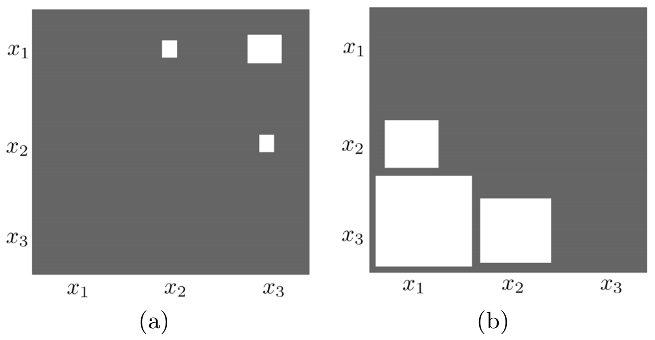


Figure 1: Figure 1(a) displays the count matrix with the contests won by each of the 3 items x_1, x_2 and x_3 after their ranking $\{r_1 = 1, r_2 = 2, r_3 = 3\}$ is converted to pairwise counts using the rank difference method. A count is displayed in each (x_i, x_j) entry if $r_i < r_j$, and the size of the square represents the count magnitude. Figure 1(b) shows the same matrix for the ranking $\{r_1 = 30, r_2 = 20, r_3 = 1\}$.

preferences are used to create a pairwise constraint matrix, and the scoring functions is then optimized to satisfy as many of these constraints as possible. The scoring function is based on a Markov Chain which makes the resulting constrained optimization problem non-convex. To solve it the authors employ a number of approximations transforming the problem into a semidefinite programming problem (SDP), which is solved using an SDP solver. The main drawback of this approach is that it is computationally very intensive and requires expensive operations such as matrix inverse and constrained optimization.

4. MULTINOMIAL PREFERENCE MODEL

In this section we develop a new score based model for pairwise preferences, the Multinomial Preference Model (MPM). A key motivating idea behind our approach is that when absolute preferences such as rankings are converted into pairwise counts using the rank difference approach described above, we interpret the resulting counts as conveying two forms of information: a binary preference, simply based on which item is ranked higher, and a confidence, based on the magnitude of the rank difference. Consider for example three items x_1, x_2 and x_3 with ranks $r_1 = 1$, $r_2 = 2$ and $r_3 = 3$ respectively. Figure 1(a) shows the resulting count matrix after these ranks are converted to pairwise preferences. Item x_1 is preferred to both x_2 and x_3 with $C(1, 2) = r_2 - r_1 = 1$ and $C(1, 3) = r_3 - r_1 = 2$, x_2 is preferred only to x_3 with $C(2, 3) = r_3 - r_2 = 1$, and x_3 is not preferred to any item. Note that preference $\{x_1 \succ x_3\}$ where both items are at the extremes of the ranking has the largest rank difference and consequently the biggest count.

Now consider the second example with partial ranking $r_1 = 30$, $r_2 = 20$ and $r_3 = 1$ yielding the pairwise count matrix shown in Figure 1(b). Comparing this with the previous example we see that the preference $\{x_3 \succ x_1\}$ with items at the extremes of the ranking also has the highest count, however in this case we are significantly more certain of it. The count $C(3, 1) = 29$ is considerably higher than the highest count from the previous example, strongly indicating that x_3 should be placed above x_1. The two examples demonstrate how large values of $C(i,j)$ may be interpreted as providing more evidence to conclude that $x_i \succ x_j$ is correct.

In MPM we model the count matrix C as an outcome of multiple draws from the joint consensus distribution Q over pairwise preferences defined by the scores. For instance in the second example above after observing C we can infer that $P(x_3 \succ x_1)$ should have the most mass under Q. We use B to denote the random variable distributed as Q. A draw from Q can be represented as a vector b_{ij} of length $M*(M-1)$ (all possible pairs), with 1 on the entry corresponding to preference $\{x_i \succ x_j\}$ and zeros everywhere else, i.e., a one-hot encoding. Given S we define the consensus distribution as follows:

Definition 1. *The consensus distribution* $Q = \{P(B = b_{ij}|S)\}_{i \neq j}$ *is a collection of pairwise probabilities* $P(B = b_{ij}|S)$, *where* $P(B = b_{ij}|S) = \frac{\exp(s_i - s_j)}{\sum_{k \neq l} \exp(s_k - s_l)}$.

Q defines a multinomial distribution over pairwise preferences. Parametrization through S controls the shape of Q, lending considerable flexibility in distributions over preferences, which can be tailored to many different problems. To generate the observed aggregated counts C we assume that T independent samples are drawn from Q where $T = \sum_{i \neq j} C(i,j)$ so that:

$$C(i,j) = \sum_{t=1}^{T} I[B = b_{ij}^{(t)}]$$

where $I[B = b_{ij}^{(t)}]$ is 1 if preference $\{x_i \succ x_j\}$ was sampled on the t'th draw and 0 otherwise. Under this model the probability of the observed counts is given by:

$$\begin{aligned} P(C|S) &= \frac{T!}{\prod_{i \neq j} C(i,j)!} \prod_{i \neq j} P(B = b_{ij}|S)^{C(i,j)} \\ &= \frac{T!}{\prod_{i \neq j} C(i,j)!} \prod_{i \neq j} \left(\frac{\exp(s_i - s_j)}{\sum_{k \neq l} \exp(s_k - s_l)} \right)^{C(i,j)} \end{aligned} \quad (5)$$

Note that in MPM the pairwise probabilities depend on the entire item set X and the observed counts matrix is modeled jointly. The magnitude of the score s_i is directly related to the count $C(i,j)$. When the scores are fitted via maximum likelihood the gradient of the log probability with respect to s_i is given by:

$$\begin{aligned} &\frac{\partial \log(P(C|S))}{\partial s_i} = \\ &\left(\sum_j C(i,j) - \sum_j C(j,i) \right) - T \left(\frac{\partial \log(\sum_{k \neq l} e^{s_k - s_l})}{\partial s_i} \right) \end{aligned} \quad (6)$$

Note that when x_i is strongly preferred to other items the first term in Equation 6 will be large leading to an increase in s_i. This will in turn raise the probability of preferences where x_i beats the other items. Raising the probability for some preferences must simultaneously lower it for others since the probabilities always sum to 1. The second term, the derivative of the partition function, accounts for this. The scores thus compete with each other and the ones with the most positive/negative evidence get pushed to the extremes. This is exactly the effect we wanted to achieve because it will allow us to accurately model the count matrices as illustrated by the toy examples above. In contrast with MPM, in the Bradley-Terry model there is no joint interaction amongst scores and pairs are modeled independently so a single preference is sufficient to push the score to infinity.

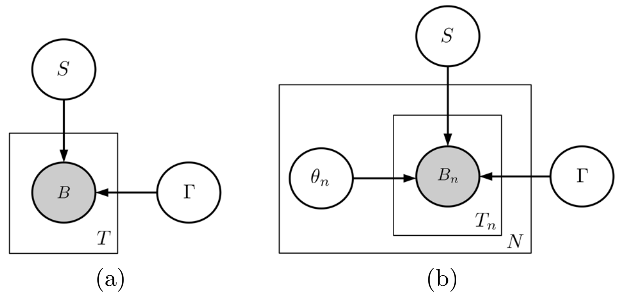


Figure 2: Graphical model representation of MPM and its θ extension.

4.1 Incorporating Prediction Confidence

In the base MPM model it is difficult to judge the model's *confidence* for a given score combination. Aside from the relative score magnitudes, it is hard to measure the uncertainty associated with the score assigned to each item and the aggregate ranking that the scores impose. Such a measure can be very useful during inference and can influence the decision process. For instance, it can be used to further filter and/or reorder the items in the aggregate ranking. Moreover, for problems where the accuracy is extremely important, the recommender system can inform the user if the produced ranking has high/low degree of uncertainty.

To address this problem we introduce a set of "variance" parameters $\Gamma = \{\gamma_1, ..., \gamma_M\}$, $\gamma_i > 0 \ \forall i$. Each γ_i models the uncertainty associated with the score s_i inferred for the item x_i. The consensus distribution now becomes:

$$P(B = b_{ij}|S, \Gamma) = \frac{\exp((s_i - s_j)/(\gamma_i + \gamma_j))}{\sum_{k \neq l} \exp((s_k - s_l)/(\gamma_k + \gamma_l))} \quad (7)$$

Note that the probability of x_i beating x_j decreases (increases) if the variance for *either* x_i or x_j increases (decreases). Through Γ we can effectively express the variance over the preferences for each item x_i and translate this variance into uncertainty over pairwise probabilities. Moreover, measures such as the average γ, $\bar{\gamma} = \frac{1}{M} \sum_{i=1}^{M} \gamma_i$, can be used to infer the variance for the entire aggregate ranking produced by the model.

In this setting γ's can either be learned in combination with scores via maximum likelihood or set using some update rule. The generative process for MPM with both S and Γ parameters is shown in Figure 2(a).

4.2 Modeling Deviations from the Consensus

The assumption in MPM that the preferences generated by the N agents are independent and identically distributed is likely to be false in many domains. Often one would expect to find preferences which either completely or partially deviate from the general consensus. For example in collaborative filtering most people tend to like popular movies such as *Harry Potter* and *Forrest Gump*, but in almost all cases one can find a number of outlier users who would give these movies low ratings. Assuming that the preferences of the outliers have the same distribution as the consensus, as is done in the base MPM model, can skew the aggregation especially if the outliers are severe.

To introduce the notion of outliers into our model we define an additional set of "adherence" parameters $\Theta = \{\theta_1, ..., \theta_N\}$, $\theta_n \in [0, 1]$. Here we assume that each agent n has its own distribution over preferences Q_n whose adherence to the global consensus distribution Q (see Definition 1) is described by θ_n. Associated with each agent n is a random variable $B_n \sim Q_n$, where we define Q_n as:

$$Q_n = \{P(B_n = b_{ij}|S, \Gamma, \theta_n)\}_{i \neq j}$$
$$P(B_n = b_{ij}|S, \Gamma, \theta_n) = \frac{\exp(\theta_n(s_i - s_j)/(\gamma_i + \gamma_j))}{\sum_{k \neq l} \exp(\theta_n(s_i - s_j)/(\gamma_i + \gamma_j))} \quad (8)$$

Note that if $\theta_n = 0$, Q_n becomes a uniform distribution indicating that the preferences of the n'th agent deviate completely from the consensus (is an outlier), and will not be modeled by it. Values between 0 and 1 indicate different degrees of agreement, with $\theta_n = 1$ indicating complete agreement. Hence, by introducing θ_n we make the model robust, allowing it to control the extent to which each agent's preferences are modeled by the scores, effectively eliminating the outliers.

In the generative process we now assume that at each of the T draws an agent n is picked at random and a preference is generated from Q_n; Figure 2(b) demonstrates this process. Under this process the probability of the observed instance $\mathbf{C} = \{C_1, ..., C_N\}$ is given by:

$$\begin{aligned} P(\mathbf{C}|S, \Gamma, \Theta) &= \\ &= \prod_{n=1}^{N} \left[\frac{T_n!}{\prod_{i \neq j} C_n(i,j)!} \prod_{i \neq j} P(B_n = b_{ij}|S, \Gamma, \theta_n)^{C_n(i,j)} \right] \\ &= \prod_{n=1}^{N} \left[\frac{T_n!}{\prod_{i \neq j} C_n(i,j)!} \prod_{i \neq j} \left(\frac{e^{\theta_n(s_i - s_j)/(\gamma_i + \gamma_j)}}{\sum_{k \neq l} e^{\theta_n(s_k - s_l)/(\gamma_k + \gamma_l)}} \right)^{C_n(i,j)} \right] \end{aligned} \quad (9)$$

where $T_n = \sum_{i \neq j} C_n(i, j)$ is the total number of preferences generated by agent n. The preferences are modeled by a mixture of N multinomials that share the same score vector S but differ in the adherence parameter θ_n. Both S and Θ can be efficiently learned by maximizing the log likelihood, and the consensus ranking can then be obtained by sorting the scores.

As noted above, in many preference aggregation problems the input typically consists of several preference instances $\{\mathbf{C}^{(\ell)}\}$, and the goal is to infer a separate set of scores $S^{(\ell)}$ and variances $\Gamma^{(\ell)}$ for each instance ℓ. The log likelihood of the entire corpus under the model is given by:

$$\mathcal{L}(\{\mathbf{C}^{(\ell)}\}|\{S^{(\ell)}\}, \{\Gamma^{(\ell)}\}, \Theta) =$$
$$\log \prod_{\ell} \prod_{n=1}^{N} \left[\frac{T_n^{(\ell)}!}{\prod_{i \neq j} C_n^{(\ell)}(i,j)!} \prod_{i \neq j} P(B_n^{(\ell)} = b_{ij}|S^{(\ell)}, \Gamma^{(\ell)}, \theta_n)^{C_n^{(\ell)}(i,j)} \right] \quad (10)$$

Here Θ is shared across the instances and the original MPM model is recovered by setting $\Theta \equiv \mathbf{1}$. When two of the three parameters $\{S^{(\ell)}, \Gamma^{(\ell)}, \Theta\}$ are fixed it is not difficult to show that $\mathcal{L}$ is concave with respect to third parameter. Therefore simple gradient descent can be used to efficiently find globally optimal setting. Furthermore, even though joint optimization is no longer convex, in the experiments we found that by using gradient descent jointly good local optimum solutions can still be found very efficiently.

4.3 Supervised Learning of Adherence Parameters

The above problem can be considered unsupervised, as the adherence parameters Θ, the consensus scores and the variances are inferred from the observed preferences. This produces a predicted ranking for a given set of observed preferences by sorting the inferred scores, without ever utilizing any known consensus rankings or relevance labels in the data.

For problems such as meta search we often have access to labeled training instances $\{\mathbf{C}^{(\ell)}\}$ for which we have the ground-truth orderings $\{L^{(\ell)}\}$ of the items $\{X^{(\ell)}\}$. In this section we describe an approach to incorporate this information into the Multinomial Preference Model. Each θ_n models the adherence of the n'th agent to the consensus. For the labeled examples the consensus is explicitly given by $L^{(\ell)}$. This allows us to exactly compute the adherence of each agent to the consensus based on the match between the preferences given by the agent and the ground truth rankings. Using this we can set θ_n to the average distance between the preferences of n'th agent and the ground truth labels:

$$\theta_n = \frac{1}{|\{\mathbf{C}^{(\ell)}\}|} \sum_{\ell} 1 - \mathcal{D}(L^{(\ell)}, C_n^{(\ell)}) \quad (11)$$

where $\mathcal{D}$ is a normalized distance metric between preferences, such as Kendall's tau. Note that as above, $\theta_n \to 1 (\to 0)$ indicates that the preferences of agent n agree with (deviate from) the consensus across the training examples.

When training examples are available the inference proceeds as follows: first training examples are used to set Θ; then keeping Θ fixed the scores and the variances are optimized on the test examples by maximizing the log likelihood.

5. META SEARCH EXPERIMENTS

For meta search aggregation problem we use the LETOR [14] benchmark datasets. These data sets were chosen because they are publicly available, include several baseline results, and provide evaluation tools to ensure accurate comparison between methods. In LETOR4.0 there are two meta search data sets, MQ2007-agg and MQ2008-agg.

MQ2007-agg contains 1692 queries with 69623 documents and MQ2008-agg contains 784 queries and a total of 15211 documents. Each query contains several lists of partial rankings of the documents under that query. There are 21 such lists in MQ2007-agg and 25 in MQ2008-agg. These are the outputs of the search engines to which the query was submitted. In addition, in both data sets, each document is assigned one of three relevance levels: 2 = highly relevant, 1 = relevant and 0 = irrelevant. Finally, each dataset comes with five precomputed folds with 60/20/20 splits for training/validation/testing. The results shown for each model are the averages of the test set results for the five folds.

The MQ2007-agg dataset is approximately 35% sparse, meaning that for an average query the partial ranking matrix of documents by search engines will be missing 35% of its entries. MQ2008-agg is significantly more sparse with the sparsity factor of approximately 65%.

Table 1: MQ2008-agg and MQ2007-agg results; statistically significant results are underlined.

	NDCG					Precision					
	N@1	N@2	N@3	N@4	N@5	P@1	P@2	P@3	P@4	P@5	MAP
MQ2008											
BordaCount	23.68	28.06	30.80	34.32	37.13	29.72	30.42	29.38	29.75	29.03	39.45
CPS-best	26.52	31.38	34.59	37.63	40.04	31.63	32.27	32.27	31.66	30.64	41.02
SVP	32.49	36.20	38.62	40.17	41.85	38.52	36.42	34.65	32.01	30.23	43.61
Bradley-Terry	38.05	39.24	40.77	41.79	42.62	44.77	39.73	36.26	33.19	30.28	44.35
Plackett-Luce	35.20	38.49	39.70	40.49	41.55	41.32	38.96	35.33	32.02	29.62	42.20
θ-MPM	**38.17**	**40.57**	**42.19**	**43.07**	**43.99**	**44.89**	**41.13**	**37.67**	**33.80**	**31.17**	**44.71**
MQ2007											
BordaCount	19.02	20.14	20.81	21.28	21.88	24.88	25.24	25.69	25.80	25.97	32.52
CPS-best	31.96	33.18	33.86	34.09	34.76	38.65	38.65	38.14	37.19	37.02	40.69
SVP	35.82	35.91	36.53	37.16	37.50	41.61	40.28	39.50	38.88	38.10	42.73
Bradley-Terry	39.88	39.86	40.40	40.60	40.91	46.34	44.65	43.48	41.98	40.95	43.98
Plackett-Luce	40.63	40.39	40.26	40.71	40.96	46.93	45.10	43.09	42.32	41.09	43.64
θ-MPM	**41.77**	**41.91**	**41.92**	**42.34**	**42.79**	**48.35**	**46.64**	**44.53**	**43.52**	**42.72**	**45.71**

The goal is to use the rank lists to infer an aggregate ranking of the documents for each query which maximally agrees with the held-out relevance levels. To evaluate this agreement we use standard information retrieval metrics: Normalized Discounted Cumulative Gain (N@K) [10], Precision (P@K) and Mean Average Precision (MAP) [2]. Given an aggregate ranking π, and relevance levels L, NDCG is defined as:

$$NDCG(\pi, L)@K = \frac{1}{G_K(L)} \sum_{i=1}^{K} \frac{2^{L(\pi^{-1}(i))} - 1}{\log(i+1)} \tag{12}$$

where $L(\pi^{-1}(i))$ is the relevance level of the document with rank i in π, and $G_K(L)$ is a normalizing constant that ensures that a perfect ordering has an NDCG value of 1. The normalizing constant allows an NDCG measure averaged over multiple queries with different numbers of documents to be meaningful. Furthermore, K is a truncation constant and is generally set to a small value to emphasize the utmost importance of getting the top ranked documents correct.

MAP only allows binary (relevant/not relevant) document assignments, and is defined in terms of average precision (AP):

$$AP(\pi, L) = \frac{\sum_{k=1}^{M} P@k * L(\pi^{-1}(k))}{\sum_{k=1}^{M} L(\pi^{-1}(k))} \tag{13}$$

where M is the number of documents; and $P@k$ is the precision at k:

$$P@k = \frac{\sum_{i=1}^{k} L(\pi^{-1}(i))}{k} \tag{14}$$

MAP is then computed by averaging AP over all queries. To compute P@k and MAP on the MQ datasets the relevance levels are binarised with 1 converted to 0 and 2 converted to 1. All presented NDCG, Precision and MAP results are averaged across the test queries and were obtained using the evaluation script available on the LETOR website.

5.1 Results

To investigate the properties of MPM we conducted extensive experiments with various versions of the model. Through these experiments we found that the supervised θ version (see Section 4.3) had the best performance; below we refer to this model as θ-MPM. Note that the training data are only used in θ-MPM to set the values of the adherence parameters Θ. Then the scores and the variances on each test query are found via maximum likelihood, and the scores are sorted to produce a predicted ranking. This is similar to the framework used by the CPS model [23] where the training data is used to estimate the ϕ parameter. In all experiments we did not take the variances into account during the sort.

We compare the results of θ-MPM against the best methods currently listed on the LETOR4.0 website,[1] namely the BordaCount model and the best of the three CPS models (combination of Mallows and Plackett-Luce models) on each of the MQ datasets. We also compare with the Bradley-Terry and Plackett-Luce models, as well as the singular value decomposition based method SVP [8]. These models cover most of the primary leading approaches in rank aggregation research. The Bradley-Terry model is fit using the same count matrices C_n that are used for MPM.

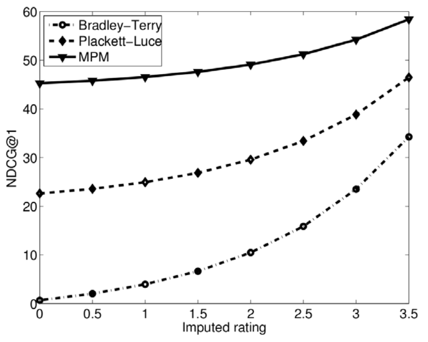
For all models we found that 100 steps of gradient descent was enough to obtain the optimal results. To avoid constrained optimization we reparametrized the variance parameters as $\gamma_i = \exp(\beta_i)$ and optimized β_i instead. This reparametrization was done for all the reported experiments. Inference with MPM is extremely fast: a MATLAB implementation took ~ 0.8 (~ 0.005 seconds per query) to make a full pass through Fold 1 (156 queries, 2874 documents) of the MQ2008-agg dataset, and ~ 4.0 seconds (~ 0.012 seconds per query) to make a full pas through Fold 1 (336 queries, 13652 documents) of the MQ2007-agg dataset.

The results for MPM together with the baselines on MQ2008-agg and MQ2007-agg datasets are shown in the top and bottom halves of Table 1 respectively. For each data set we conducted a paired T-test between θ-MPM and the best baseline at each of the 5 truncations for NDCG and precision as well as MAP, the statistically significant results at the 0.05 level are underlined. From the table we see that the θ-MPM models significantly outperforms the baselines on the MQ2007-agg dataset on both NDCG and MAP metrics. On MQ2008-agg θ-MPM is also the best model, significantly improving over the baselines on truncations 2-4 for NDCG and 2,3,5 for Precision.

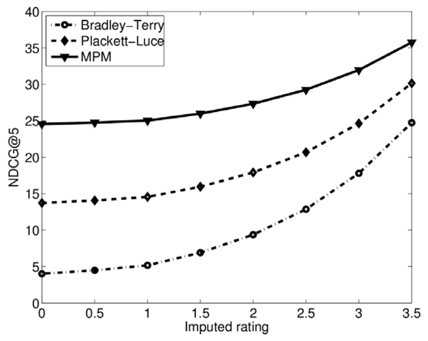
[1]research.microsoft.com/en-us/um/beijing/projects/letor/

Table 2: NDCG results for the MovieLens data set, for each user the missing ratings are filled using the probabilistic matrix factorization model; statistically significant results are underlined.

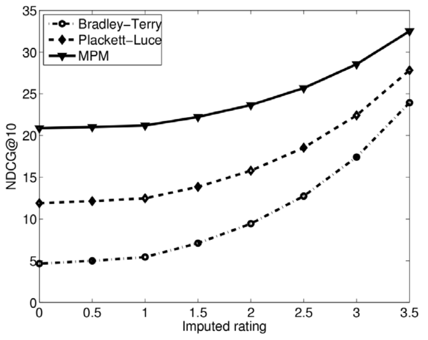
	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10
Bradley-Terry	40.09	36.00	35.20	34.96	34.49	34.40	31.63	32.08	32.46	32.35
Plackett-Luce	**69.56**	54.17	48.97	46.58	44.89	43.44	42.50	41.25	40.64	40.03
MPM	69.15	**54.29**	**49.72**	**46.98**	<u>**45.52**</u>	<u>**44.13**</u>	<u>**43.25**</u>	<u>**42.62**</u>	<u>**42.04**</u>	<u>**41.57**</u>



(a)



(b)



(c)

Figure 4: Plots of NDCG at truncations 1, 5 and 10; in this setting all the missing ratings were repeatedly imputed by one of the constants shown on the x-axis and the rankings given by each method were evaluated using NDCG (Equation 15). All the differences are statistically significant.

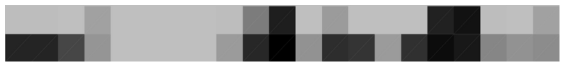


Figure 3: Top row: normalized Θ, found by the supervised procedure outlined in Section 4.3, for training Fold 1 of MQ2007-agg. Bottom row: learned Θ on the same Fold. Here white = 1 and black = 0.

Figure 3 shows the adherence parameters Θ set based on the labeled training examples, together with the one learned in an unsupervised fashion by doing gradient descent on both S and Θ simultaneously. From the figure we see many similarities in the two vectors, indicating that the model is able to capture the notion of ”outliers” which correlates closely with the training labels. There are however a number of differences, such as the first three components being switched from on to off in the learned Θ. In our experiments we found that setting Θ using the training labels consistently produced better performance.

6. COLLABORATIVE FILTERING EXPERIMENTS

For collaborative filtering experiments we used the MovieLens dataset, a collection of 100,000 ratings (1-5) from 943 users on 1682 movies. This data set was chosen because it provides demographic information such as age and occupation for each user, as well as movie information such as genre, title and release year. Each user in this data rated at least 20 movies but the majority of ratings for each movie are missing and the rating matrix is more than 94% sparse. We formulate the preference aggregation as follows: given users' ratings the goal is to come up with a single ranking of the movies that accurately summarizes the majority of user preferences expressed in the data. This ranking could be used as an initial recommendation for a new user who has not provided any ratings yet, as well as in a summary page. Note that the aggregation can be further personalized by only aggregating over users that share similar demographic and/or other factors with the target user.

To convert ratings into preferences we can either sort them (resolving ties), to obtain a partial ranking for each user, or use the pairwise method to obtain the count matrices C_n, where $C_n(i,j) = (l_{ni} - l_{nj})I[l_{ni} > l_{nj}]$ if movies x_i and x_j were rated by user n and 0 otherwise. We use the sort method for the permutation based Plackett-Luce model and use the rating difference method for the pair based Bradley-Terry and MPM models.

In collaborative filtering and in most other applications the primary goal of aggregation is to recommend items to a new or existing user. Items ranked in the top few positions are of particular interest because they are the ones that will typically be shown to the user. Intuitively a top ranked item should have ratings from many users (*high support*) and most who rated it should prefer it to other items (*strong preference*). Consequently NDCG suggests itself as a good metric to evaluate the rankers for this problem because of its emphasis on the top ranked items and the truncation level structure. Unlike meta search the ground truth ratings are not available for most collaborative filtering data. To get around this problem we complete the rating matrix by imputing the missing ratings for every user. We investigate two methods of imputing the ratings: a user independent method, where all the missing ratings are filled in by the same value, and a user dependent method, where for every user n the missing ratings are predicted by a probabilistic matrix factorization model (PMF) [24]. The reason for choosing PMF was that it has shown excellent performance

Table 3: Top 5 and bottom 5 movies found by each model. For each movie the table shows the number of users that rated it (#u) and the total number of pairwise contests that the movie won (#won) and lost(#lost) across all users.

Bradley-Terry	#u	#won	#lost	Plackett-Luce	#u	#won	#lost	MPM	#u	#won	#lost
Pather Panchali	8	1431	89	Shawshank Red.	283	32592	5943	Star Wars	583	49290	10112
Wallace & Gromit	67	7448	962	Wallace & Gromit	67	7448	962	Raiders of the L.	420	40057	10644
Casablanca	243	26837	4633	Usual Suspects	267	30779	6666	Godfather	413	36531	8040
Close Shave	112	11219	1963	Star Wars	583	49290	10112	Silence of the L.	390	38192	9125
Rear Window	209	22590	4513	Wrong Trousers	118	13531	2291	Shawshank Red.	283	32592	5943
.				.				.			
.				.				.			
.				.				.			
Children of Corn	19	62	3161	Barb Wire	30	462	5507	Cable Guy	106	3469	14377
Lawnmower Man 2	21	129	3868	Robocop 3	11	125	2535	Striptease	67	1347	9909
Free Willy 3	27	171	3912	Gone Fishin'	11	123	1098	Very Brady	93	3353	12509
Kazaam	10	128	2041	Highlander III	16	881	2826	Jungle2Jungle	132	2375	11086
Best of the Best 3	6	33	1445	Ready to Wear	18	289	3785	Island of Dr.	57	1176	9415

on collaborative filtering tasks such as the Netflix challenge. After completing the rating matrix we compute the NDCG value for every user by sorting the items according to scores:

$$NDCG(\pi, L_n)@K = \frac{1}{G_K(L_n)} \sum_{i=1}^{K} \frac{2^{L_n(\pi^{-1}(i))} - 1}{\log(i+1)} \qquad (15)$$

Here π is the aggregated ranking obtained by sorting the items according to scores, and $\pi^{-1}(i)$ is the index of the item with rank i in π; L_n is a (completed) vector of ratings for user n. G_K is the normalizing constant and represents the maximum DCG value that could be obtained for n:

$$G_K(L_n) = \sum_{i=1}^{K} \frac{2^{L_n(\sigma^{-i}(i))} - 1}{\log(i+1)} \qquad (16)$$

where σ is a permutation of L_n with the ratings sorted from largest to smallest. In this form if for a given user n an item in position i in π has a rating lower than the rating $L_n(\sigma^{-1}(i))$ of the i'th highest rated item by n, the corresponding term in the NDCG summation will decrease exponentially with the difference between $L_n(\sigma^{-1}(i))$ and $L_n(\pi^{-1}(i))$. We use this metric (averaged across all users) to evaluate the performance of the models.

6.1 Results

We compare the results of MPM to the Bradley-Terry and Plackett-Luce models, the two best baselines on the meta search task. For all models we found that 100 steps of gradient descent was enough to reach convergence.

The NDCG results from the user dependent rating imputation method are shown in Table 2. From this table we see that MPM outperforms the best baseline, Plackett-Luce, on all truncations except 1 with statistically significant gains at truncations 5-10. This is likely due to the fact that in MPM the score magnitude is directly related to the number of observations. The model has a strong bias to put movies with a large number of observations at the extremes of the ranking.

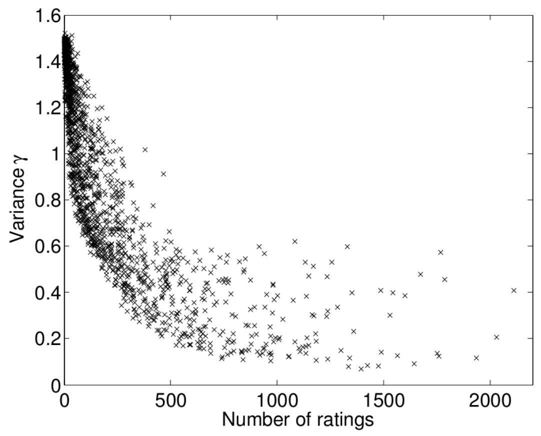
The NDCG plots for the user independent rating imputation method are shown in Figure 4. The plots show NDCG at truncations 1, 5 and 10 for the three methods, when each of the values in $\{0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5\}$ was used to fill the missing ratings. Here, the value 3.5 was chosen as the upper boundary because it is the average rating for the MovieLens data set. A number of studies have shown that users tend to rate items that they like so the average of the observed ratings is typically significantly higher than the average of the the unobserved ones [19]. From the figure we see that MPM significantly outperforms both the Bradley-Terry and Plackett-Luce models. The differences are especially large when low values are imputed for the missing ratings. This indicates that the Bradley-Terry and Plackett-Luce models place items that were rated by very few users (*low support*) at the top of the list. This causes the imputed ratings to dominate the numerator in the NDCG summation making the results very sensitive to the magnitude of the imputed rating.

This effect can also be observed from Table 3 which shows the top and bottom 5 movies generated by each model together with statistics on the number of users that rated each movie and the number of pairwise contests lost and won by the movie (summed across all users). For a given user n and movie i with rating l_{ni} we find the number of pairwise wins by counting the number of pairs $\{i, j\}$ with $l_{ni} > l_{nj}$; losses are found in a similar way. From the table we see that the Bradley-Terry model places the movie *Pather Panchali* at the top of the list. This movie is only rated by 8 out of 943 users and even though most users who rated it preferred it to other movies (#lost is low) there is still very little evidence that this movie represents the top preference for the majority of users. Due to its pairwise independence assumption the Bradley-Terry model is always likely to place movies with few ratings near the top/bottom of the list.

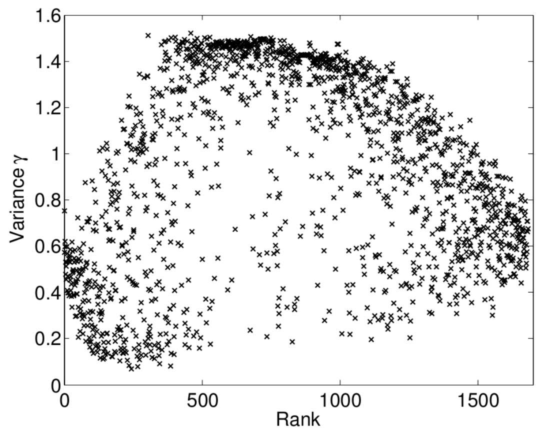
The Plackett-Luce model partially fixes this problem by considering items jointly, and places the frequently rated movie *Shawshank Redemption* first. However the model does not fully eliminate the problem, placing the very infrequently rated *Wallace & Gromit* (also ranked second by Bradley-Terry) in the second spot. Part of the reason for this comes from the fact that the Plackett-Luce is a permutation based model and as such cannot model the strength of preferences, treating the preferences given for example by ratings $\{5, 2, 1\}$ the same as $\{5, 4, 3\}$.

On the other hand for the Multinomial Preference Model we see that the position of the item is related to both the number of observed preferences and the strength of those preference. The top three movies are all rated by more than 400 users and are strongly preferred by the majority of those users.

A more severe pattern can be observed for the bottom 5



(a)



(b)

Figure 5: 5(a) shows the number of ratings versus the learned variance γ_i for each movie x_i. 5(b) shows the rank for each movie obtained after sorting the scores versus the learned γ_i.

movies. Both Bradley-Terry and Plackett-Luce place movies rated by less than 30 users in the bottom 5 positions labeling them the worst movies in the entire data set. This selection has very little evidence in the data and has a high probability of being wrong if more ratings are collected. For MPM all of the bottom 5 movies are rated by more than 50 users with 3 out of 5 movies rated by more than 90 users.

In addition to the retrieval accuracy we investigated the properties of the learned variance parameters γ. Figure 5(a) shows the learned variances together with the number of ratings for each movie. Note that the variance is inversely proportional to the number of ratings so as the number of ratings increases the model becomes increasingly more certain in the preferences decreasing the variance. In Figure 5(b) we plot γ against the aggregate rank for each movie. The general pattern is clear: the variance decreases towards the extremes of the ranking, indicating that the model is more certain in the movies that are placed near the top and near the bottom of the aggregate ranking. As shown above, this is due to the fact that the movies at the extremes of the ranking have many comparisons, allowing accurate inference of strong negative or positive preferences.

The plot however, also shows outliers, which are the movies placed in the middle of the aggregrate ranking with low variance/high confidence. After further inspection we found that each such movie had many positive as well as negative preferences. Examples of these include *Sabrina* (#u:190 #won:10190 #lost:12347), *Mrs. Doubtfire* (#u:192 #won:13251 #lost:17551) and *Ghost* (#u:170 #won:11785 #lost:14452). Note that all three movies were rated by more than 150 users and overall were neither strongly preferred nor strongly disliked. The model thus correctly placed them in the middle of the ranking with strong confidence. Moreover, note that it is impossible to express this confidence with scores alone since all the movies in the middle of the ranking have similar scores. The variances thus provide additional information about the decisions made by the model during the aggregation, which could be very useful for post processing and evaluation.

7. CONCLUSION AND FUTURE WORK

We have introduced a new probabilistic model over preferences based on a multinomial generative process. Preferences over items are expressed through real valued scores resulting in a convex optimization problem during inference which can be solved efficiently with standard gradient based techniques. Modeling the general partial pairwise preferences makes the model applicable to a wide range of preference aggregation problems. Empirically we have shown that our approach outperforms existing preference aggregation methods on two unrelated problems: meta search and collaborative filtering.

Future work includes developing supervised extensions of the model that can more directly utilize the labeled training data available in problems such as meta search. Another interesting direction is to investigate how the learned variances can be used to improve the final ranking. Finally, we also plan to explore mixtures of the MPM distributions where each mixing component is parametrized by its own set of scores. The mixture could be trained to learn different user preference types and used for personalized recommendation.

8. REFERENCES

[1] J. A. Aslam and M. Montague. Models for metasearch. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 276–284, 2001.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Information Retrieval*. Addison-Wesley, 1999.

[3] R. Bradley and M. Terry. Rank analysis of incomplete block designs. I. The method of paired comparisons. *Biometrika*, 39:324–345, 1952.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the International Conference on Machine Learning*, pages 89–96, 2005.

[5] P. Dangauthier, R. Herbrich, T. Minka, and T. Graepel. TrueSkill through time: Revisiting the

history of chess. In *Proceedings of the Neural Information Processing Systems*, 2007.

[6] A. E. Elo. *The rating of chess players: Past and present.* Acro Publishing, 1978.

[7] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 301–312, 2003.

[8] D. F. Gleich and L.-H. Lim. Rank aggregation via nuclear norm minimization. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 60–68, 2011.

[9] J. Guiver and E. Snelson. Bayesian inference for Plackett-Luce ranking models. In *Proceedings of the International Conference on Machine Learning*, pages 377–384, 2009.

[10] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 41–48, 2000.

[11] X. Jiang, L.-H. Lim, Y. Yao, and Y. Ye. Statistical ranking and combinatorial hodge theory. *Mathematical Programming*, 127:203–244, 2011.

[12] A. Klementiev, D. Roth, and K. Small. Unsupervised rank aggregation with distance-based models. In *Proceedings of the International Conference on Machine Learning*, pages 472–479, 2008.

[13] G. Lebanon and J. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In *Proceedings of the International Conference on Machine Learning*, pages 363–370, 2002.

[14] T. Liu, J. Xu, W. Xiong, and H. Li. LETOR: Benchmark dataset for search on learning to rank for information retrieval. In *ACM SIGIR Workshop on Learning to Rank for Information Retrieval*, 2007.

[15] Y.-T. liu, T.-Y. Liu, T. Qin, Z.-M. Ma, and H. Li. Supervised rank aggregation. In *Proceedings of the International conference on World Wide Web*, pages 481–489, 2007.

[16] T. Lu and C. Boutilier. Learning Mallows models with pairwise preferences. In *Proceedings of the International Conference on Machine Learning*, 2011.

[17] R. D. Luce. *Individual choice behavior: A theoretical analysis.* Wiley, 1959.

[18] C. L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.

[19] B. M. Marlin, R. S. Zemel, and S. T. Roweis. Unsupervised learning with non-ignorable missing data. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 222–229, 2005.

[20] D. Mase. A penalized maximum likelihood approach for the ranking of college football teams independent of victory margins. *The American Statistician*, 57:241–248, 2003.

[21] M. Meila, K. Phadnis, A. Patterson, and J. Bilmes. Consensus ranking under the exponential model. In *Proceedings of the Uncertainty in Artificial Intelligence Conference*, 2007.

[22] R. Plackett. The analysis of permutations. *Applied Statistics*, 24:193–302, 1975.

[23] T. Quin, X. Geng, and T.-Y. Liu. A new probabilistic model for rank aggregation. In *Proceedings of the Neural Information Processing Systems*, pages 681–689, 2010.

[24] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Proceedings of the Neural Information Processing Systems*, volume 20, 2008.

[25] L. L. Thurstone. The method of paired comparisons for social values. *Journal of Abnormal and Social Psychology*, 21:384–400, 1927.