Forty-Ninth Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
September 28 - 30, 2011

Ranking: Compare, don't Score

Ammar Ammar, Devavrat Shah
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
{ammar, devavrat}@mit.edu

Abstract—**The need to rank items based on user input arises in many practical applications such as elections, group-decision making and recommendation systems. The primary challenge in such scenarios is to decide a global ranking based on partial preferences provides by users. The standard approach to address this challenge is to ask users to provide explicit numerical ratings (cardinal information) of a subset of items. The main appeal of such an approach is the ease of aggregation. However, the rating scale as well as the individual ratings are often arbitrary and may not be consistent from one user to another. A more natural alternative to numerical ratings requires users to compare pairs of items (ordinal information). In contrast to cardinal information, such comparisons provide an "absolute" indicator of the user's preference. However, it is often hard to combine or aggregate comparisons to obtain a consistent global ranking.**

In this work, we provide a tractable framework for utilizing comparison data as well as first-order marginal information for the purpose of ranking. We treat the available information as partial samples from an unknown distribution over permutations. Using the Principle of Maximum Entropy, we devise a concise parameterization of distribution consistent with observations using only $O(n^2)$ parameters, where n is the number of items in question. We propose a distributed, iterative algorithm for estimating the parameters of the distribution. We establish the correctness of the algorithm as well as identify the rate of convergence explicitly. Using the learnt distribution, we provide efficient approach to (a) learn the mode of the distribution using 'maximum weight matching', (b) identification of top k items, and (c) an aggregate ranking of all n items. Through evaluation of our approach on real-data, we verify effectiveness of our solutions as well as the scalability of the algorithm.

I. Introduction

Judging, rating or ranking objects is omnipresent: whether it be restaurants in a city, movies on Netflix, books on Amazon, candidates interviewed for faculty positions or papers submitted to Allerton conference. In all such instances, a global ranking of objects is achieved based on the inputs about partial rankings provided by a large number of people.

The current practice is to seek input in terms of scores, e.g. assign between 1 to 5 stars to a restaurant/movie or score between 1 to 10 for a paper. The key advantage of seeking such quantitative input is that it is easy to achieve global aggregation: in Allerton, for example, each paper may receive scores from, say 3 UIUC reviewers, between 1 to 10; the average of these scores will lead to the global ranking of all the submitted papers to assist in making the final acceptance/rejection decisions.

On the flip side, the key disadvantage stems from the fact that scores are *relative*: the score of 6, for example, may be interpreted differently by different individuals and further, the same individual may score objects differently depending upon the context, for example the order in which s/he reviewed the assigned papers. While one may argue that it could be possible to correct for such "biases" for reviewers about whom we have known history, such an approach is somewhat ad-hoc and even not feasible when the ratings are obtained anonymously.

An alternative approach of seeking input[1], which we advocate in this paper, is qualitative: for example, ask reviewers explicitly to compare the papers they reviewed (score assignments do not necessarily achieve this as there could be a tie and in the context of anonymous ratings, this is totally different from quantitative rating). The key advantage of seeking such information is that it is definitely more *absolute*: when two individuals say they like A over B, they do mean the same; or a reviewer is likely to compare two papers in the same way despite the order in which they review them. It is no surprise that many polling sites (e.g. Washington Post [2], WFNX [1]) have started using such interfaces to collect information. Further in many settings data in naturally available in such form, for example customers revealing their preferences among items on display in a shop by purchasing one of them, cf. [14].

The key challenge with qualitative information arises in the aggregating phase, due to possible contradictions: for three items A, B and C, we could receive inputs A preferred over B, B preferred over C and C preferred over A from different individuals. Such seeming conflicts have created challenges for aggregation over centuries starting the celebrated work of Condorcet [9]; also see work on impossibility of existence of rankings pioneered by Arrow [5]. Now unlike the standard setting of the, so called ranked elections considered in the literature following the works [9] [5], in our context, we have access to partial ranking information of objects: in general, we have a fraction of population comparing a given pair of objects unlike in the standard ranked election literature where each individual provides complete ranking.

The main contribution of this paper lies in a proposal of novel method for aggregation of such partial (qualitative) ranking information to come up with a global ranking. The key insight is to view the collected data as the partial information about an underlying distribution over complete orderings of all objects. Therefore, the problem of aggregation reduces to

[1]We believe that in an ideal system, inputs of both forms should be obtained for better decision making. In this paper, we focus on qualitative inputs primarily to understand what sorts of information, on its own, does it contain.

(a) learning the distribution over rankings or permutation of objects that is (near) consistent with the observed data, and (b) using this distribution to obtain the final ranking of objects. Our approach to resolve (a) consists of finding the distribution with maximal entropy (near) consistent with the observed data; and (b) by means of a novel aggregation mechanism using the distributional information. Before we describe our contributions in further detail, we quickly recall related work.

Related work. The question of learning distribution over permutations from partial or limited information has been well studied in the recent literature. Notably, in the work of Huang, Guestrin and Guibas [16], the task of interest is to infer the most likely permutation of identities of objects that are being tracked through noisy sensing by maintaining distribution over permutations. To deal with the 'factorial blowup', authors propose to maintain only the first-order marginal information of the distribution (essentially corresponding to certain Fourier co-efficients), then use the Fourier inversion formula to recover the distribution and subsequently predict its mode as the likely assignment. In the work by Jagabathula and Shah [17], authors took a different approach to the same problem where they proposed to learn the distributed over permutations by finding the sparsest distribution consistent with the observed partial information. Finally, this approach was further extended and integrated with the decision making in the context of revenue management in work by Farias, Jagabathula and Shah [14]. None of these works, however deals with the question of aggregation or achieving ranking. While the finding mode is a candidate for such a ranking, it is 'fragile' as we shall discuss soon. It should also be noted that, through maximum entropy distribution learning, we are trying to be maximally unconstrained subject to observed data, unlike the above cited approaches which implicitly or explicitly impose additional constraints (e.g. sparsity).

The task of ranking objects or assigning scores has been of great interest over the past decade or so with similar concerns. There is a long list of works, primarily in the context of bipartite ranking, including the RankBoost by Freund et al. [15], label ranking by Dekel et al. [11], Crammer and Singer [10], Shalev-Shwartz and Singer [22] as well as analytic, learning results on bipartite ranking including those of Agarwal et al. [3], Usunier et al. [23] and Rudin and Schapire [21]. The algorithm that will be closest to one of the proposal is the p-norm push algorithm by Rudin [20] which uses ℓ_p norm of information to achieve ranking. As we shall establish, the ranking used by [2], [1] to aggregate comparison data is equivalent to the ℓ_1 norm of the underlying distribution. The empirical results suggest that our proposed use of 'exponentially' weight ranking provides better aggregation. Indeed, to obtain such rank aggregation, it is necessary to learn the underlying distribution.

The maximum entropy approach for learning distribution is a classical now (including the work of Boltzman). The maximum entropy (max-ent) distribution, a member of an appropriate exponential distribution family, is maximum likelihood estimation of the parameters in that family (cf. see [24]). Indeed, the use of exponential family distribution over rankings has been around for more than few decades now (cf. see [12, Chapter 9]). We provide a careful analysis of a stochastic sub-gradient algorithm for learning the parameters of this max-entropy distribution. This algorithm is distributed and iterative. It is directly build upon such an algorithm used in [19] for distributed wireless scheduling. It is worth taking note of use of maximum entropy distribution over permutation based on given marginal information to learn the "missing" marginals by Agrawal et al [4] in the context of parimutuel betting.

Our contributions. The primary contribution of this paper is use of distribution over permutations as means to reach rank aggregation from collection of partial preferences.

The stochastic gradient algorithm for learning the max-ent distribution is derived from [19], however the proof is different (and simpler). It provides explicit rate of convergence in the context of both (a) pair-wise comparisons, and (b) first-order marginals. Using the standard MCMC and their known mixing time bounds, our analysis suggests that the computation time scales exponentially in n and polynomially in n respectively for the pair-wise comparisons and first-order marginals respectively. Two remarks are in order. First, the result for first-order marginals also suggest a distributed scheduling algorithm for input-queued switch with polynomial time learning complexity (unlike exponential for wireless network model). Second, the standard stochastic approximation based approaches cf. [8] does not apply as is (due to compactness of domain related issue).

The distribution, thus learnt, is used in multiple ways. First, for application like the object-tracking [16], the goal is to learn the maximum likelihood assignment (or mode) of the distribution. Given the form of the max-ent distribution (exponential family), it reduces to solving maximum weight matching problem in a bipartite graph with weights induced by thus learnt parameters. Now for the first-order representation of the permutations, this is an easy instance of the network-flow problem (can be solve, for example, using belief propagation [6]). For pair-wise comparisons representation, the problem is not known to be solvable in polynomial time (we believe its hard). We propose a simple randomized scheme that is a 2-approximation of it.

We propose a heuristic for mode computation as well that bypasses the step of learning the max-ent parameters but uses directly the available partial preference data. Such a heuristic, for example, can speed up computation of [16] drastically. Somewhat curiously, we show that this heuristic is first-order approximation of the mode finding of the max-ent distribution.

The mode is a likely candidate for rank aggregation. However it is not "robust": there could be many permutations with similar probability that are drastically different from each other. In contrast, if we are interested in, say identifying top k items out of n items, the approach we propose is as follows: for each item, identify the marginal probability (which is readily available as input with marginal data, but need to be evaluated for comparison information using maximum entropy distribution) that it is ranked to positions between 1 to k. Use this as a score for each item and take the top k as per this score.

More generally, we introduce a natural exponential function based relaxation of such a top k ranking to obtain ranking over all elements that uses a single parameter, say Θ. The p-norm based ranking (similar to the p-norm push [20]) is a closely related approach. However, as we establish using empirical results that the exponential function based Θ-ranking provide better answer than p-norm based ranking with $p = 1$.

We perform experimental study to understand (a) the goodness of the ranking produced by our algorithm, and (b) the scalability of the algorithm. While goodness of ranking is a *subjective* notion, we use available online polls (by Washington Post) to show that compared to the standard algorithm used in practice, our algorithm provides more detailed answers. We conducted a real-time experiment during an the 150 year celebration event of MIT, where for an entire day various people continually entered their inputs through multiple web-interfaces and our algorithm maintained the rankings (which are very good) in an online manner. This is testimonial to the scalability of our algorithm especially given that it was operating on an in-expensive machine. The iterative and distributed nature of our algorithm is especially useful in its ability to cope with scale.

II. Model and Problem Statement

Model: We consider a universe of n available items, $\mathcal{N} = \{1, 2, ..., n\}$. Each user has preference order, represented as permutation, over these n items. Specifically, if σ is the permutation, the user prefers item i over j if $\sigma(i) < \sigma(j)$. We assume that there is a distribution, say μ, over the space of permutations of n items, S_n, that defines the collective preferences of the entire user population.

Data: We consider scenarios where we have access to partial or limited information about μ. Specifically, we shall restrict our attention to two popular types of data: first-order ranking and comparisons. Each of these two types correspond to some sort of marginal distribution of μ as follows:

First-order marginals: For any $1 \leq i, k \leq n$, the fraction of population that ranks item i as their kth choice is the first-order marginal information for distribution μ. Specifically,

$$m_{ik} \overset{\triangle}{=} \mathsf{P}_\mu[\{\sigma(i) = k\}] = \sum_{\sigma \in S_n} \mu(\sigma) \mathbb{I}_{\{\sigma(i)=k\}} \tag{1}$$

where $\mathbb{I}_{\{E\}}$ denotes the indicator variable for event E. Collectively, we have the $n \times n$ matrix $[m_{ij}]$ of the first-order marginals, that we shall denote by M. This is precisely the type of information that was maintained for tracking agents in the framework introduced by Huang, Guestrin and Guibas [16].

Comparison Data: For any $1 \leq i, j \leq n$, the fraction of population that prefers item i over item j is the comparison marginal information. Specifically,

$$c_{ij} \overset{\triangle}{=} \mathsf{P}_\mu[\{\sigma(i) < \sigma(j)\}] = \sum_{\sigma \in S_n} \mu(\sigma) \mathbb{I}_{\{\sigma(i)<\sigma(j)\}}. \tag{2}$$

Collectively, we have access to the $n \times n$ matrix $[c_{ij}]$ of comparison marginals, denoted by C. Such data is available through customer transactions in any business, cf. [14].

Two remarks. First, while we assume m_{ik} (resp. c_{ij}) available for all i, k (resp. i, j), if only a subset of it is available, the algorithm with that information works equally well. Second, we shall assume that $m_{ik} \in (0, 1)$ for all i, k (resp. $c_{ij} \in (0, 1)$ for all i, j).

Goal: Roughly speaking, the goal is to aggregate data of type M or C to obtain a ranking of objects of interest. Specifically, we are interested in (a) finding 'most likely' (or mode) ranking, (b) the top k objects, and (c) the aggregate ranking over all n objects.

To address these questions, our approach is to first learn a distribution that is consistent with the observation (M or C), and then use this learnt distribution to answer the questions (a), (b) and (c). In principle, there could be multiple distributions that are consistent with the observed data (M or C, assuming it is generated by a consistent underlying unknown distribution). As mentioned earlier, we shall choose the max-ent distribution that is consistent with the observed data.

III. The Maximum Entropy Model

Formally, the observations M or C impose constraint that the distribution of choice should belong to class $\mathcal{M}$:

$$\sum_{\sigma \in S_n} \mu(\sigma) \mathbb{I}_{\{\sigma(i)=k\}} \ = \ m_{ik}, \quad \forall i, k \in \mathcal{N} \tag{3}$$

or class $\mathcal{C}$:

$$\sum_{\sigma \in S_n} \mu(\sigma) \mathbb{I}_{\{\sigma(i)<\sigma(j)\}} \ = \ c_{ij}, \quad \forall i, j \in \mathcal{N} \tag{4}$$

with the the normalization and non-negativity constraints in both cases.

$$\sum_{\sigma \in S_n} \mu(\sigma) \ = \ 1, \quad \mu(\sigma) \ \geq \ 0, \quad \forall \sigma \in S_n. \tag{5}$$

$\mathcal{M}$ (resp. $\mathcal{C}$) is non-empty only if M (resp. C) is generated by a distribution over S_n to begin with. We shall assume that such is the case. In practice, clearly this is not a reasonable assumption. The algorithm that we shall present is based on the solving the Lagrangian dual of an appropriate optimization problem in which the constraints imposed by M (resp. C) are "dualized". Therefore, by construction such algorithm is robust.

Now $|S_n| = n!$ and the data of type M (resp. C) imposes $O(n^2)$ constraints. Therefore, there could be multiple solutions. The max-ent distribution suggest that we choose the one that has maximal entropy in the class $\mathcal{M}$ (resp. $\mathcal{C}$). Philosophically, we follow this approach since we wish to utilize the information provided by the data and nothing else, i.e. we do not wish to impose any additional structure beyond what data suggests. It is also well known that such a distribution provides maximum likelihood estimation over certain class of exponential family distributions (cf. [24]). In effect, the goal is to find the distribution that solves the following optimization:

$$\max_{\nu} \quad H_{\mathrm{ER}}(\nu) \stackrel{\triangle}{=} -\sum_{\sigma\in S_N} \nu(\sigma)\log\nu(\sigma)$$
$$\nu \in \mathcal{M} \quad \text{or} \quad \mathcal{C}. \tag{6}$$

It can be checked that the Lagrangian dual of this problem is as follows (since all entries of M, C in $(0,1)$): let λ_{ik} be the dual variables associated with marginal consistency constraint for $\mathcal{M}$ in (3). Then, the dual takes the following form:

$$\max_{\lambda} \sum_{i,k} \lambda_{ik} m_{ik} - \log\Big(\sum_{\sigma}\exp\Big(\sum_{ik}\lambda_{ik}\mathbb{I}_{\{\sigma(i)=k\}}\Big)\Big) \tag{7}$$

It can be shown that this is a strictly concave optimization and has a unique optimal solution. Let it be $\lambda^* = [\lambda^*_{ik}]$. Then the corresponding primal optimal solution of (6) (with $\mathcal{M}$) is given by

$$\mu(\sigma) \propto \exp\Big(\sum_{i,k\in\mathcal{N}} \lambda^*_{ik}\cdot\mathbb{I}_{\{\sigma(i)=k\}}\Big). \tag{8}$$

Similarly, for the comparison data, the dual optimization takes the form

$$\max_{\lambda} \sum_{i,j} \lambda_{i<j} c_{ij} - \log\Big(\sum_{\sigma}\exp\Big(\sum_{ij}\lambda_{i<j}\mathbb{I}_{\{\sigma(i)<\sigma(j)\}}\Big)\Big), \tag{9}$$

and the optimal primal of (6) given optimal dual $\lambda^* = [\lambda^*_{i<j}]$ is

$$\mu(\sigma) \propto \exp\Big(\sum_{i\neq j\in\mathcal{N}} \lambda^*_{i<j}\cdot\mathbb{I}_{\{\sigma(i)<\sigma(j)\}}\Big). \tag{10}$$

As can be seen, in either case the maximum entropy distribution is parameterized by at most n^2 parameters, which is the same as the degrees of freedom of the received data. We shall, with abuse of notation, use $F(\lambda)$ to represent the objective of both Lagrangian dual optimization problems (7) and (9).

A. A Subgradient Algorithm

Here we describe an iterative, distributed sub-gradient algorithm that solves the dual optimization problems (7), (9). First, we describe an idealized procedure that calls certain oracle that estimates marginals of distribution from exponential family. We can, in general, only hope to estimate these marginals approximately. Therefore, the main result that we state is for a sub-gradient algorithm based on such an approximate oracle. In a later section, we shall describe how to design such an approximate oracle in a distributed manner along with its associated computational cost.

Algorithm 1 MaxEnt Estimation: Using Ideal Oracle

```
Require: Ranking data $m_{ik}$   $\forall i,k$.
1: Initialize: $\lambda^0_{ik} = 0\ \forall i,k$.
2: for $t = 1 \to T$ do
3:     $\lambda^{t+1}_{ik} \leftarrow \lambda^t_{ik} + \frac{1}{\sqrt{t}}\Big(m_{ik} - \mathsf{E}_{\lambda^t}[\mathbb{I}_{\{\sigma(i)=k\}}]\Big)$
       ($\mathsf{E}_{\lambda^t}[\mathbb{I}_{\{\sigma(i)=k\}}]$ is provided by an oracle)
4: end for
5: Choose $\tau \in \{1,\dots,T\}$ at random so that $\mathsf{P}(\tau = t) \propto 1/\sqrt{t}$
6: return $\lambda^\tau$
```

Here, $\mathsf{E}_{\lambda^t}[\mathbb{I}_{\{\sigma(i)=k\}}] = \sum_{\sigma\in S_n}\mathbb{P}_{\lambda^t}(\sigma)\mathbb{I}_{\{\sigma(i)=k\}}$ where

$$\mathbb{P}_{\lambda^t}(\sigma) = \frac{1}{Z(\lambda^t)}\exp\Big(\sum_{i,k}\lambda^t_{ik}\mathbb{I}_{\{\sigma(i)=k\}}\Big),$$

with normalizing constant (partition function) $Z(\lambda^t) = \sum_{\sigma\in S_n}\exp\Big(\sum_{i,k}\lambda^t_{ik}\mathbb{I}_{\{\sigma(i)=k\}}\Big)$. We shall use an (randomized) estimation, $\tilde{\mathsf{E}}_{\lambda^t}(i,k) = \tilde{\mathsf{E}}_{\lambda^t}[\mathbb{I}_{\{\sigma(i)=k\}}]$, of $\mathsf{E}_{\lambda^t}[\mathbb{I}_{\{\sigma(i)=k\}}]$ so that the error vector $\mathbf{e}(t) = [\mathbf{e}_{ik}(t)]$ with $\mathbf{e}_{ik}(t) = \mathsf{E}_{\lambda^t}[\mathbb{I}_{\{\sigma(i)=k\}}] - \tilde{\mathsf{E}}_{\lambda^t}(i,k)$ is appropriately small enough. We state the following result about the convergence of this algorithm.

Theorem 1. *Suppose the sub-gradient algorithm uses approximate estimation $\tilde{E}.(\cdot,\cdot)$ such that $\|\mathbf{e}(t)\|_1 \le \frac{1}{A(t)+\|\lambda^*\|_\infty+\|\lambda^*\|_2^2}$, where $A(t) = \sum_{s=1}^{t} 1/\sqrt{s}$ and λ^* is a solution of the optimization problem. Then, for any $\gamma > 0$, for choice of $T = \Theta\Big(\epsilon^{-2-\delta}\Big(n^2+\|\lambda^*\|_\infty+\|\lambda^*\|_2^2\Big)^{2+\delta}\Big)$, we have*

$$\mathsf{E}\Big[F(\lambda^\tau)\Big] \ge F(\lambda^*) - \epsilon,$$

where $F(\cdot)$ is the objective of dual optimization (7)*. The identical result holds for the comparison information* (9)*.*

B. An Approximate Oracle

Theorem 1 relies on existence of an oracle that can produce an estimation of marginals approximately with appropriate accuracy for each time step t. Here we describe such an oracle. We shall restrict our description to the Markov Chain Monte Carlo (MCMC) based oracle. In principle, one may use heuristic like Belief Propagation to estimate these marginals instead of MCMC (of course, this may lead to loss of performance guarantee).

Now the computation of marginals requires computing $\mathbb{P}_{\lambda^t}(\sigma)$ for any $\sigma \in S_n$. From its form, the basic challenge is in computing the partition function $Z(\lambda^t)$. The partition function $Z(\lambda^t)$ is the same as computation of permanent of a non-negative valued matrix $A = [A_{ik}]$ where $A_{ik} = \mathbf{e}^{\lambda_{ik}}$. In an amazing work, Jerrum, Sinclair and Vigoda [18] have design Fully Polynomial Time Randomized Approximation Scheme (FPRAS) for computing permanent of any non-negative valued matrix. That is, $Z(\lambda^t)$ (hence $\mathbb{P}_{\lambda^t}(\sigma)$) can be computated within multiplicative accuracy $(1\pm\varepsilon)$ in time polynomial in $1/\varepsilon, n, \log(1/\delta)$ with probability at least $1-\delta$. Therefore, it follows that the desired guarantee in Theorem 1 can be provided for all timesteps (using union bound) with probability at least $1 - 1/n$ within polynomial in n building upon the algorithm of [18].

For the case of comparison information, however no such FPRAS algorithm for computing partition function is known. Therefore, we suggest a simple MCMC based algorithm and provide the obvious (exponential) bound for it. To that end, define $W_\lambda(\sigma) = \sum_{i,k \in \mathcal{N}} \lambda_{ik} \cdot \mathbb{I}_{\{\sigma(i)<\sigma(j)\}}$, and construct a Markov chain, $\mathfrak{M}(\lambda)$, whose state space is the set of all permutations, S_n, and whose transitions from a given state σ to a new state σ' are given as follows:

```
1: With probability $\frac{1}{2}$ let $\sigma' = \sigma$.
2: Otherwise, construct $\sigma'$ as follows:
     ◦ Choose two elements $i$ and $j$ uniformly at random;
       set $\tilde{\sigma}(i) = \sigma(j)$, $\tilde{\sigma}(j) = \sigma(i)$ and $\tilde{\sigma}(k) = \sigma(k)$ for
       all $k \neq i, j$.
     ◦ Set $\sigma' = \tilde{\sigma}$ with probability $\min\{1, \exp\big(W_\lambda(\tilde{\sigma}) - W_\lambda(\sigma)\big)\}$; else set $\sigma' = \sigma$.
```

Using this Markov chain, we can estimate the desired $\mathsf{E}_\lambda[\mathbb{I}_{\{\sigma(i)<\sigma(j)\}}]$ as follows: starting from any initial state, run the Markov chain for T_m steps and then record the state of the Markov chain, say σ^{T_m}. If $\sigma^{T_m}(i) < \sigma^{T_m}(j)$, then record 1 else record 0. Repeat this for S times and obtain the empirical average of the recorded $0/1$ values. Declare this as the estimation of $\mathsf{E}_\lambda[\mathbb{I}_{\{\sigma(i)<\sigma(j)\}}]$. Indeed, one simultaneously obtains such estimation for all i, j. We have the following bound on T_c:

Theorem 2. *The above stated Markov chain has stationary distribution μ^* so that*

$$\mu^*(\sigma) \propto exp\Big(W_\lambda\Big).$$

Let $\mu(t)$ be the distribution of the Markov chain after t steps starting from any initial condition. Then for any given $\delta > 0$, there exists

$$T_c = \Theta\Big(exp\Big(\Theta\big(n^2\|\lambda\|_\infty + n\,log\,n\big)\Big)\,log\,\frac{1}{\delta}\Big),$$

such that for $t \geq T_c$,

$$\Big\|\frac{\mu(t)}{\mu^*} - 1\Big\|_{2,\mu^*} < \delta,$$

where $\|\cdot\|_{2,\mu}$ is the χ^2 distance.

Now the total variation distance between $\mu(t)$ and μ^* is smaller than the χ^2 distance between them. Therefore, by Theorem 2, it follows that the error in estimation of the $\mathbb{P}_\lambda(\sigma)$ using $\mu(t)$ will be at most δ. From Chernoff's bound, by selecting S (mentioned above) to be $O(\delta^{-2}\log n)$ (with large enough constant), it will follow that the estimated empirical marginals for all i, j components must be within error $O(\delta)$ with probability $1 - 1/\text{poly}(n)$. Given that increment in each component of λ as part of the sub-gradient algorithm is $O(\sqrt{T})$ by time T, from Theorem 1, it follows that the $\|\lambda\|_\infty = O\big((n + \|\lambda^*\|_\infty + \|\lambda^*\|_2^2)^{1+\gamma}\big)$ (for any choice of $\gamma > 0$ in Theorem 1). Finally, the smallest δ required in Theorem 1 is inverse polynomial in n, ϵ, from above discussion it follows that the overall cost of the approximate oracle required for the comparisons effectively scales exponential in $n^{3+\gamma}$ (ignoring other smaller order terms).

IV. Ranking

Thus far, we have described algorithm for estimating distribution consistent with the observed data by finding appropriate max-ent distribution. Now we describe how this learnt distribution can be used for (a) find mode of the distribution, (b) top-k ranked elements, and (c) overall aggregate ranking of all n objects using what we call the Θ-ranking.

A. Mode = Max Weight Matching

The mode or the most likely element of distribution is the $\sigma \in S_n$ with the maximal probability. Since log of the probability of σ with 'dual' variables (or parameters of the exponential family) is proportional to $\sum_{i,k}\lambda_{ik}\mathbb{I}_{\{\sigma(i)=k\}}$ for first-order marginal and $\sum_{i,j}\lambda_{i<j}\mathbb{I}_{\{\sigma(i)<\sigma(j)\}}$ for comparison marginals, it boils down to finding

$$\sigma^* \in \arg\max_{\sigma \in S_n}\Big(\sum_{i,k}\lambda_{ik}\mathbb{I}_{\{\sigma(i)=k\}}\Big) \qquad \text{first-order marginal,} \tag{11}$$

$$\sigma^* \in \arg\max_{\sigma \in S_n}\Big(\sum_{i,j}\lambda_{i<j}\mathbb{I}_{\{\sigma(i)<\sigma(j)\}}\Big) \qquad \text{comparison.} \tag{12}$$

The problem in (11) is equivalent to the following maximum weight matching problem: consider an $n \times n$ complete bipartite graph with edge between node i on left and node k on right having weight λ_{ik}. A matching is a subset (of size n) edges so that no two edges are incident on same vertext. Let weight of the matching is the summation of the weights of the edge chosen by it. Then the maximum weight matching in this graph is precisely solving (11). This is a well known instance of the classical network flow problem and has strongly polynomial time algorithms [13]. It also allows for distributed iterative algorithm for finding it including the auction algorithm of Bertsekas [7] and the recently popular (max-product) belief propagation [6]. Thus, overall finding the mode of the distribution for the case of first-order marginal is *easy* and admits distributed algorithmic solution.

The problem in (12) is also equivalent to a combinatorial problem with the space of objects being the matchings. However, it does not admit the nice representation as above. One way to represent the matchings in comparison form is $n \times n$ matrices, say $B = [B_{ij}]$ with (a) each entry B_{ij} being $+1$ or -1 for all $1 \leq i, j \leq n$, (b) for all $1 \leq i, j \leq n$, $B_{ij} + B_{ji} = 0$ (anti-symmetric), and (c) if $B_{ij} = B_{jk} = 1$, then $B_{ik} = 1$ for all $1 \leq i, j, k \leq n$. The goal is to find B so that $\sum_{ij} B_{ij}\lambda_{i<j}$ is maximized. It is not clear if this is an easy problem. We describe a simple 2-approximation algorithm:

choose L permutatations uniformly at random, compute their weights (defined as per (12)) and select the one with maximal weight among these L permutations. For L large enough, this is essentially with $1/2$ weight of the maximum weight. This requires λ to have all non-negative components. This is not an issue since given the structure of the permutations (each having equal number comparisons, $\sigma(i) < \sigma(j)$, correct) and hence an affine transformation of λ by vector with all components being same constant does change the distribution. Therefore, in principle, we could require the subgradient algorithm to be restricted to the non-negative domain (projected verison). The formal statement about this algorithm is stated below.

Theorem 3. *Let $\lambda = [\lambda_{i<j}]$ be non-negative vector. Let OPT be the maximum of $\sum_{ij} \lambda_{i<j} \mathbb{I}_{\{\sigma(i)<\sigma(j)\}}$ among all permutation $\sigma \in S_n$. Then in the above described randomized algorithm, if we choose $L \geq \frac{1}{2\delta} \ln \frac{1}{\epsilon}$, then*

$$\mathbb{P}\Big[W(\hat{\sigma}) < \frac{1}{2}(1-\delta)OPT\Big] < \epsilon$$

1) Heuristic for Mode: Here we describe a heuristic for finding mode without requiring the intermediate step of finding the max-ent parameter λ. We describe it for first-order marginal. Declare the solution of the following optimization as the mode:

$$\max \sum_{i,k} m_{ik} \mathbb{I}_{\{\sigma(i)=k\}}.$$

That is, in place of λ_{ik}, use m_{ik}. The intuition is that λ_{ik} is higher if m_{ik} is and vice versa. While there is no direct relation between this heuristic and mode of the max-ent approximation; we state the following which establishes the heuristic to be a 'first-order' approximation.

Theorem 4. *For $\lambda = [\lambda_{ik}]$ in small enough neighborhood of* $\mathbf{0} = [0]$,

$$m_{ik} \approx \frac{1}{n} + \frac{1}{n-1} \lambda_{ik}.$$

B. Top-k Ranking

Here interest is in finding top k ranked objects (the favorites). One approach to declare the top k ranked objects in the mode. But quality of such a ranking is "peaky" or "non-robust". This is because it is possible for many other permutations to have probability close to that of the mode. This is especially true for "flat" distributions.

To avoid this, we propose a natural way to select favorites. Intuitively, if an object is ranked among top k positions by a large fraction (probability-wise) of the permutations in the distribution, then it ought to be among favorites. This suggests that for each object i, we should associate score $S_k(i)$, defined as

$$S_k(i) = \mathbb{P}_\lambda[\sigma(i) \leq k],$$

where $\mathbb{P}_\lambda$ is the max-ent distribution we learnt. Indeed, the above score is nothing but $\sum_{\ell \leq k} m_{i\ell}$ for the first-order marginals (and hence we do not need to learn max-ent); of course, it is inferred from max-ent distribution learnt for the case of comparison. Now declare the top k objects with highest scores as per $S_k(\cdot)$ as the result of top k.

C. Θ-Ranking

To obtain an entire ranking of all objects, again, mode is not robust. Building on intuition behind top-k ranking, the basic premise is that the objects that are ranked higher more frequently should be getting higher ranking. With this in mind, let us define scores for objects as follows. For any monotonically strictly increasing non-negative function $f : \mathbb{N} \to [0, \infty]$, define score $S_f(i)$ for object i as

$$S_f(i) = \sum_{k=1}^{n} f(n-k) \mathbb{P}_\lambda(\sigma(i) = k). \tag{13}$$

The choice of $f(x) = x^p$ assigns the pth norm of the distribution of $\sigma(i)$ as score to object i. Again, by definition, the score can be computed directly from given data using the first-order marginals (no need to find the max-ent). However for comparisons, again, it is necessary to know the max-ent distribution to compute ranking with the exceptional case of $p = 1$. Specifically for $p = 1$, the score of object i equals $n - \sum_{j \neq i} c_{ij}$. This equivalence is explained in Section V.

The exponential function, $f_\theta(x) = \exp(\theta x)$ for $\theta > 0$ effectively caputres the combined effect of all p-norms. Therefore, we propose, what we call the Θ-ranking, based on the following scores: for $\theta > 0$,

$$S_\theta(i) = \sum_{k=1}^{n} \exp(-\theta k) \mathbb{P}_\lambda(\sigma(i) = k). \tag{14}$$

Indeed, by selection $\theta \approx \ln k$, the scores are effectively capturing the occurence of objects in top k positions only; and for θ near 0 they are capturing the effect of lower p moments more prominently. As we shall see, intermediate choice of θ give effective ranking for various empricial scenarios considered in the next section.

V. Empirical Evaluation

In this section, we provide an empirical evaluation of our maximum-entropy approach we have proposed. In particular, we address the issues of ranking quality, and scalability through two experiments. Our evaluation is two fold: first, we compare our approach with a simple comparison-based ranking method, which does not require learning a distribution over permutations. We show that this method is equivalent to a specific, subset of our $\theta-$ranking approach. The juxtaposition suggests that there is more to be gained from using the "higher-order" instances using $\Theta-$ranking. We also provide an empirical comparison between the two methods, and highlight the gains in the quality of the ranking. Secondly, we discuss a large scale implementation of the ranking algorithm.

A. Why Learn the Maximum Entropy

One of the main concerns in designing a voting or ranking system is that of simplicity. Put another way, if one is going to abandon a simple practice (e.g. score-based ranking), then the alternatives should be comparably simple, or at least justify any additional complexity.

Recall that one form of data that is available for our algorithm is that of pairwise comparisons (i.e. $\mathbb{P}\big[\sigma(i) < \sigma(j)\big]$). If we think of these comparisons as the average number of wins in repeated pairwise matches, then ranking can be seen as an attempt to decide who won more matches on average. A score that reflects this intuition is given by:

$$S(i) = \frac{1}{n-1} \sum_{j \neq i} \Pr[\sigma(i) < \sigma(j)]$$

This score is the probability of winning a match given a uniform prior on the opponents. Interestingly, this quantity is equivalent, within a multiplicative factor, to the score dicussed in the Θ-ranking section, given by:

$$S_p(i) = \sum_{k=1}^{n} (n-k)^p \cdot \Pr[\sigma(i) = k]$$

with $p = 1$. More concretely, we have the following lemma, which we prove in section **??**.

Lemma 1. *Given the definition of $S(i)$ and $S_1(i)$ above, we have*

$$S_1(i) = \frac{1}{n-1} \sum_{j \neq i} \mathbb{P}\big[\sigma(i) < \sigma(j)\big]$$

This suggests that higher values of p, and their natural extension in the form of $\theta-$ranking should enable us to incorporate more information in our ranking, thus justifying the additional complexity incurred by learning the maximum-entropy distribution.

To confirm this intuition, we apply the two methods to a survey conducted by the Washington Post in late 2010, under the title "Who had the Worst Year in Washington?". As a part of the survey, participants were asked to compare people, institutions, or ideologies two at a time. Chosen results from the two methods are provided below.

Pairwise Wins	**θ-Ranking ($\theta = 4$)**
Liberalism	Progressives
Progressives	Liberalism
Blue Dog Democrats	President Obama
Nancy Pelosi	Blue Dog Democrats
President Obama	Nancy Pelosi
Taxpayers	Michael Steele
Michael Steele	Taxpayers
Michele Bachman	Mitch McConnell
Mitch McConnell	Michele Bachmann

While the results in both cases are close, we point out that our algorithm is consistent in maintaining the winners and losers (e.g. progressives and Michele Bachmann). Furthermore, our algorithm seems to do a better job at capturing subtle differences. For instance, if we look at the last two items, we note that Michele Bachmann, given her successful political career so far, is a more likely candidate for the top winner. Another interesting example is that of Michael Steele, who made some controversial comments prior to the survey, and taxpayers. These observations, are of course speculative, and further investigation is underway.

B. *System Implementation and Scalability*

In this "experiment", we implemented a voting/survey tool that enables a large number of participants to vote on any number of items in real time. By doing so, we had the following two questions in mind: in addition o being theoritically interesting, can our algorithm be applied in real time? is comparison based voting practical and simple enough for adoption? through this exeperiment, we believe the answer to both questions to be affirmative.

Our tool was implemented as a web application with a backend ranking engine. To avoid a "cognitive overload", or boring the participants, our interface provides each participant with a set of 9 items selected from a larger universe. The user then can express his preference over that set by drag-and-dropping items in the right order. This ordering is then converted into pairwise comparisons and sent to the ranking engine, and the resulting ranking is sent back for instant display.

Our tool was installed in voting booths that were made available to the visitors of the MIT 150 open house. Voting categories included movies, actors, musicians, atheletes, among others, and the results at any time were continuously displayed on a large screen. The participation was impressive, and the feedback was mostly positive, which makes us believe that adopting comparison as form of voting is worth a serious consideration.

References

[1] The absolute top 101 songs. http://new.wfnx.com/supplements/2011/topsongs/battle/.

[2] Who had the "worst year in washington"? http://voices.washingtonpost.com/thefix/worst-week-in-washington/worst-year-in-washington.html.

[3] S. Agarwal, T. Graepel, R. Herbrich, S. Har-Peled, and D. Roth. Generalization bounds for the area under the roc curve. *Journal of Machine Learning Research*, 6(1):393, 2006.

[4] S. Agrawal, Z. Wang, and Y. Ye. Parimutuel betting on permutations. *Internet and Network Economics*, pages 126–137, 2008.

[5] K.J. Arrow. *Social choice and individual values*. Number 12. Yale Univ Pr, 1963.

[6] M. Bayati, D. Shah, and M. Sharma. Max-product for maximum weight matching: Convergence, correctness, and lp duality. *Information Theory, IEEE Transactions on*, 54(3):1241–1251, 2008.

[7] D.P. Bertsekas. The auction algorithm: A distributed relaxation method for the assignment problem. *Annals of Operations Research*, 14(1):105–123, 1988.

[8] V.S. Borkar. *Stochastic approximation: a dynamical systems viewpoint*. Cambridge Univ Pr, 2008.

[9] M. Condorcet. *Essay sur l'application de l'analyse de la probabilité des decisions: Redues et pluralité des voix*. l'Imprimerie Royale, 1785.

[10] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292, 2002.

[11] O. Dekel, C. Manning, and Y. Singer. Log-linear models for label ranking. *Advances in neural information processing systems*, 16, 2003.

[12] P. Diaconis. *Group representations in probability and statistics*, volume 11. JSTOR, 1988.

[13] J. Edmonds and R.M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

[14] V. Farias, S. Jagabathula, and D. Shah. A data-driven approach to modeling choice. *Advances in Neural Information Processing Systems*, 22:504–512, 2009.

[15] Y. Freund, R. Iyer, R.E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *The Journal of Machine Learning Research*, 4:933–969, 2003.

[16] J. Huang, C. Guestrin, and L. Guibas. Efficient inference for distributions on permutations. *Advances in neural information processing systems*, 20:697–704, 2008.

[17] S. Jagabathula and D. Shah. Inferring rankings under constrained sensing. *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[18] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004.

[19] L. Jiang, D. Shah, J. Shin, and J. Walrand. Distributed random access algorithm: scheduling and congestion control. *Information Theory, IEEE Transactions on*, 56(12):6182–6207, 2010.

[20] C. Rudin. The p-norm push: A simple convex ranking algorithm that concentrates at the top of the list. *The Journal of Machine Learning Research*, 10:2233–2271, 2009.

[21] C. Rudin and R.E. Schapire. Margin-based ranking and an equivalence between adaboost and rankboost. *The Journal of Machine Learning Research*, 10:2193–2232, 2009.

[22] S. Shalev-Shwartz and Y. Singer. Efficient learning of label ranking by soft projections onto polyhedra. *The Journal of Machine Learning Research*, 7:1567–1599, 2006.

[23] N. Usunier, M.R. Amini, and P. Gallinari. A data-dependent generalisation error bound for the auc. In *Proceedings of the ICML 2005 Workshop on ROC Analysis in Machine Learning*. Citeseer, 2005.

[24] M.J. Wainwright and M.I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.