Mach Learn (2010) 80: 189–211
DOI 10.1007/s10994-010-5176-9

Preference-based learning to rank

Nir Ailon · Mehryar Mohri

Received: 15 March 2009 / Accepted: 1 November 2009 / Published online: 29 April 2010

Abstract This paper presents an efficient preference-based ranking algorithm running in two stages. In the first stage, the algorithm learns a preference function defined over pairs, as in a standard binary classification problem. In the second stage, it makes use of that preference function to produce an accurate ranking, thereby reducing the learning problem of ranking to binary classification. This reduction is based on the familiar QuickSort and guarantees an expected pairwise misranking loss of at most twice that of the binary classifier derived in the first stage. Furthermore, in the important special case of bipartite ranking, the factor of two in loss is reduced to one. This improved bound also applies to the regret achieved by our ranking and that of the binary classifier obtained.

Our algorithm is randomized, but we prove a lower bound for any deterministic reduction of ranking to binary classification showing that randomization is necessary to achieve our guarantees. This, and a recent result by Balcan et al., who show a regret bound of two for a deterministic algorithm in the bipartite case, suggest a trade-off between achieving low regret and determinism in this context.

Our reduction also admits an improved running time guarantee with respect to that deterministic algorithm. In particular, the number of calls to the preference function in the reduction is improved from $\Omega(n^2)$ to $O(n \log n)$. In addition, when the top k ranked elements only are required ($k \ll n$), as in many applications in information extraction or search engine design, the time complexity of our algorithm can be further reduced to $O(k \log k + n)$. Our algorithm is thus practical for realistic applications where the number of points to rank exceeds several thousand.

Keywords Learning to rank · Machine learning reductions · ROC

Editors: Sham Kakade and Ping Li.

N. Ailon (✉) · M. Mohri
Computer Science Faculty, Technion – Israel Institute of Technology, Haifa 32000, Israel
e-mail: nailon@cs.technion.ac.il

M. Mohri
Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, NY 10012, USA

1 Introduction

The learning problem of ranking arises in many modern applications, including the design of search engines, information extraction, and movie recommendation systems. In these applications, the ordering of the documents or movies returned is a critical aspect of the system.

The problem has been formulated within two distinct settings. In the *score-based setting*, the learning algorithm receives a labeled sample of pairwise preferences and returns a *scoring function* $f : U \to \mathbb{R}$ which induces a linear ordering of the points in the set U. Test points are simply ranked according to the values of f for those points. Several ranking algorithms, including RankBoost (Freund et al. 2003; Rudin et al. 2005), SVM-type ranking (Joachims 2002), and other algorithms such as PRank (Crammer and Singer 2001; Agarwal and Niyogi 2005), were designed for this setting. Generalization bounds have been given in this setting for the pairwise misranking error (Freund et al. 2003; Agarwal et al. 2005), including margin-based bounds (Rudin et al. 2005). Stability-based generalization bounds have also been given in this setting for wide classes of ranking algorithms both in the case of bipartite ranking (Agarwal and Niyogi 2005) and the general case (Cortes et al. 2007a, 2007b).

A somewhat different two-stage scenario was considered in other publications starting with (Cohen et al. 1999), and later (Balcan et al. 2007, 2008), which we will refer to as the *preference-based setting*. In the first stage of that setting, a preference function $h : U \times U \mapsto [0, 1]$ is learned, where values of $h(u, v)$ closer to one indicate that u is ranked above v and values closer to zero the opposite. The preference function h is typically assumed to be the output of a classification algorithm trained on a sample of labeled pairs, and can be for example a convex combination of simpler preference functions as in Cohen et al. (1999). A crucial difference with the score-based setting is that, in general, the preference function h may not induce a linear ordering. The relation it induces may be non-transitive, thus we may have for example $h(u, v) = h(v, w) = h(w, u) = 1$ for three distinct points u, v, and w. To rank a test subset $V \subseteq U$, in the second stage, the algorithm orders the points in V by making use of the preference function h learned in the first stage. The subset ranking set-up examined by Cossock and Zhang (2006), though distinct, also bears some resemblance with this setting.

This paper deals with the preference-based ranking setting just described. The advantage of this setting is that the learning algorithm is not required to return a linear ordering of all points in U, which may be impossible to achieve faultlessly in accordance with a general possibly non-transitive pairwise preference labeling. This is more likely to be achievable exactly or with a better approximation when the algorithm is requested instead, to supply a linear ordering, only for limited subsets $V \subseteq U$.

When the preference function is obtained as the output of a binary classification algorithm, the preference-based setting can be viewed as a reduction of ranking to classification. The second stage specifies how the ranking is obtained using the preference function.

Cohen et al. (1999) showed that in the second stage of the preference-based setting, the general problem of finding a linear ordering with as few pairwise misrankings as possible with respect to the preference function h is NP-complete. The authors presented a greedy algorithm based on the tournament degree, that is, for a given element u, the difference between the number of elements it is preferred to versus the number of those preferred to u. The bound proven by the authors, formulated in terms of the pairwise disagreement loss l with respect to the preference function h, can be written as $l(\sigma_{\text{greedy}}, h) \leq 1/2 + l(\sigma_{\text{optimal}}, h)/2$, where $l(\sigma_{\text{greedy}}, h)$ is the loss achieved by the permutation σ_{greedy} returned by their algorithm and $l(\sigma_{\text{optimal}}, h)$ the one achieved by the optimal permutation

σ_{optimal} with respect to the preference function h. Here, the loss l is normalized to be in the range $[0, 1]$. This bound was given for the general case of ranking, but, in the particular case of bipartite ranking, a random ordering can achieve a pairwise disagreement loss of $1/2$ and thus the bound is not informative. Note also that the algorithm can be viewed as a derandomization technique.

More recently, Balcan et al. (2008) studied the bipartite ranking problem. In this particular case, the loss of an output ranking is measured by counting pairs of ranked elements, one of which is positive and the other negative, based on some ground truth—a more formal definition of the bipartite loss function is given later. Each time a negative element appears higher in the output compared to a positive element, a penalty is added to the loss. They showed that sorting the elements of V according to the same tournament degree used by Cohen et al. (1999) guarantees a loss of at most twice the loss of the binary classifier used. They also showed that this guarantees a regret of at most $2r$ for a binary classifier with regret r.[1] However, due to the quadratic nature of the definition of the tournament degree, their algorithm requires $\Omega(n^2)$ calls to the preference function h, where $n = |V|$ is the number of objects to rank. The advantage of their algorithm is that it is deterministic.

We describe an efficient randomized algorithm for the second stage of the preference-based setting and thus for reducing the learning problem of ranking to binary classification. We guarantee, for the bipartite ranking case, an *expected* pairwise misranking regret of at most r using a binary classifier with regret r, thereby bringing down the regret factor from two to one. Compared to Balcan et al. (2008), we offer a tradeoff of low (expected) regret versus determinism. Our reduction applies, with different constants, to a broader class of ranking loss functions, admits a simple proof, and the expected running time complexity of our algorithm in terms of number of calls to a classifier or preference function is improved from $\Omega(n^2)$ to $O(n \log n)$. Furthermore, when the top k ranked elements only are required ($k \ll n$), as in many applications in information extraction or search engines, the time complexity of our algorithm can be further reduced to $O(k \log k + n)$. Our reduction and algorithm are thus practical for realistic applications where the number of points to rank exceeds several thousands. The price paid for this improvement is in resorting to randomization, but we prove a lower bound for any deterministic reduction of ranking to binary classification showing that randomization is necessary to achieve our guarantees: we give a simple proof of a lower bound of $2r$ for *any* deterministic reduction of ranking to binary classification with classification regret r. This result generalizes to all deterministic reductions a lower bound given by Balcan et al. (2008) for a specific algorithm.

To understand the low regret versus determinism tradeoff, it is interesting to consider the case of a binary classifier with an error rate of just 25%, which is quite reasonable in many applications. Assume that the Bayes error is close to zero for the classification problem and, similarly, that for the ranking problem the regret and loss approximately coincide. Then, the bound of Balcan et al. (2008) guarantees for the ranking algorithm a worst-case pairwise misranking error of at most 50%, which is the pairwise misranking error of random ranking. In this work, we give an algorithm that provably achieves an expected pairwise misranking error of at most 25%. Of course, since the algorithm is randomized, the actual error may reach 50% with some probability.

Much of our results also extend beyond the bipartite case already discussed to the general case of ranking. However, these extended results apply only to the loss and not to the regret and at the cost of a worse factor of two, instead of one. A by-product of this extension also allows us to compare our results to those given by Cohen et al. (1999).

[1] Balcan et al.'s (2008) definition of regret is based on a specific assumption that we shall discuss later.

The algorithm used by Balcan et al. (2008) to produce a ranking based on the preference function is known as sort-by-degree and has been recently used in the context of minimizing the feedback arcset in tournaments (Coppersmith et al. 2006). Here, we use a different algorithm, QuickSort, which has also been recently used for minimizing the feedback arcset in tournaments (Ailon et al. 2005; Ailon 2007). The techniques presented build upon earlier work by Ailon et al. (2005) and Ailon (2007) on combinatorial optimization problems over rankings and clustering.

The results just mentioned were already included in an earlier version of this paper (Ailon and Mohri 2008). Here, we further generalize all of these results to the case where the preference function h takes values in $[0, 1]$ and generalize the algorithm used in the second stage, QuickSort, to make use of these real-valued scores instead of just their rounded binary values. Most classification algorithms return a real-valued score that is subsequently used to determine the labels. These scores can often be interpreted as confidence scores and thus be more informative than their rounded or quantized counterparts. This is clear in the case of generative models such as logistic regression, but the scores output by discriminative learning algorithms are also often normalized and used as confidence scores. We show that the same guarantees can be provided for the loss and regret bounds in both the bipartite and the general ranking cases with the loss and regret for the preference functions defined with respect to the real-values of h and thus taking into account the confidence scores of the algorithm that produced h. This gives a strict generalization of our previous results (Ailon and Mohri 2008) since the case where the scores are rounded is a special case corresponding to a preference function $\tilde{h}\colon U \mapsto \{0, 1\}$. This generalization in fact helps simplify some of our proofs. Note that our results also generalize over those of Balcan et al. (2008) which hold for the case of a preference function taking values in $\{0, 1\}$.

The remainder of the paper is structured as follows. In Sect. 2, we introduce the definitions and notation used in future sections and introduce a general family of loss functions for ranking. Section 3 describes a simple and efficient algorithm for reducing ranking to binary classification, proves several bounds guaranteeing the quality of the ranking produced by the algorithm, and analyzes the running-time complexity of our algorithm. In Sect. 4, we derive a lower bound for any deterministic reduction of ranking to binary classification. In Sect. 5, we discuss the relationship of the algorithm and its proof with previous related work in combinatorial optimization, and discuss some key assumptions related to the notion of regret in this context.

2 Preliminaries

This section introduces several preliminary definitions necessary for the presentation of our results. In what follows, U will denote a universe of elements, e.g., the collection of all possible query-result pairs returned by a web search task, and $V \subseteq U$ will denote a small subset thereof, e.g., a preliminary list of relevant results for a given query. For simplicity of notation we will assume that U is a set of integers, so that we are always able to choose a minimal canonical element in a finite subset, as we shall do in (12) below. This arbitrary ordering should not be confused with the ranking problem we are considering.

2.1 General definitions and notation

We first briefly discuss the learning setting and assumptions made here and compare them with those of Balcan et al. (2008) and Cohen et al. (1999).

In what follows, $V \subseteq U$ represents a finite subset extracted from some arbitrary universe U, which is the set we wish to rank at each round. The notation $S(V)$ denotes the set of *rankings* on V, that is the set of injections from V to $[n] = \{1, \ldots, n\}$, where $n = |V|$. If $\sigma \in S(V)$ is such a ranking, then $\sigma(u)$ is the rank of an element $u \in V$, where lower ranks are interpreted as preferable ones. More precisely, we say that u is preferred over v with respect to σ if $\sigma(u) < \sigma(v)$. For convenience, and abusing notation, we also write $\sigma(u, v) = 1$ if $\sigma(u) < \sigma(v)$ and $\sigma(u, v) = 0$ otherwise. We let $\binom{V}{k}$ denote the collection of all subsets of size exactly k of V. We denote by $\mathrm{E}_{a,b,\ldots}[\cdot]$ the expectation taken with respect to the variables $a, b, \ldots$. To distinguish between functions taking ordered versus unordered arguments, we use the notation $F_{u_1 u_2 \ldots u_k}$ to denote k unordered arguments for a function F defined on $\binom{V}{k}$, and $F(u_1, u_2, \ldots, u_k)$ for ordered arguments.

2.2 Preference function

As with both (Cohen et al. 1999) and (Balcan et al. 2008), we assume that a preference function $h\colon U \times U \to [0, 1]$ is learned in the first learning stage. The convention is that the higher is $h(u, v)$, the more our belief that u should be preferred to v. The function h satisfies *pairwise consistency*: $h(u, v) + h(v, u) = 1$, but needs not even be transitive on three tuples (cycles may be induced). The second stage uses h to output a proper ranking σ, as we shall further discuss below. The running time complexity of the second stage is measured with respect to the number of calls to h.

2.3 Output of learning algorithm and loss function

The cost of the permutation σ output by the second stage of the algorithm is measured differently in Balcan et al. (2008) and Cohen et al. (1999). In the former, it is measured against the unknown ground truth and compared to the cost of the preference function h against the ground truth. In Cohen et al. (1999), σ is measured against the preference function h, and compared to the theoretically best ordering, that is the ordering that is the closest to h. Thus, here, h plays the role of the ground truth.

Most of our work follows the approach of Balcan et al. (2008): at each round, there is an underlying unknown ground truth which we wish the output of the learning algorithm to agree with as much as possible. The ground truth is a ranking which we denote by $\sigma^* \in S(V)$, equipped with a function ω assigning different *importance* weight to pairs of positions. The combination (σ^*, ω) we allow in this work, as we shall see below, is very expressive. It can encode in particular the standard average pairwise misranking or AUC loss assumed by Balcan et al. (2008) in a bipartite setting, but also more sophisticated ones capturing misrankings among the top k, and other losses that are close but distinct from those considered by Clémençon and Vayatis (2007).

The following general loss function L_ω measures the quality of a ranking σ with respect to a desired one σ^* using a symmetric weight function ω described below:

$$L_\omega(\sigma, \sigma^*) = \binom{n}{2}^{-1} \sum_{u \neq v} \sigma(u, v)\sigma^*(v, u)\omega(\sigma^*(v), \sigma^*(u)).$$

The sum is over all pairs u, v in the domain V of the rankings σ, σ^*. It counts the number of inverted pairs $u, v \in V$ weighed by ω, which assigns importance coefficients to pairs, based on their positions in the ground truth σ^*. The function ω must satisfy the following three natural axioms, which will be necessary in our analysis:

(P1) Symmetry: $\omega(i, j) = \omega(j, i)$ for all i, j;
(P2) Monotonicity: $\omega(i, j) \leq \omega(i, k)$ if either $i < j < k$ or $i > j > k$;
(P3) Triangle inequality: $\omega(i, j) \leq \omega(i, k) + \omega(k, j)$.

The motivation for property (P3) is as follows: if the importance of ordering correctly items in positions i, k and k, j is not very great, then, it is also not very important to correctly order the items in positions i, j.

The space of admissible importance functions ω contains many useful, well studied distance functions. Setting $\omega(i, j) = 1$ for all $i \neq j$ yields the unweighted pairwise misranking measure or the so-called Kemeny distance function.

For a fixed integer k, the following function

$$\omega(i, j) = \begin{cases} 1 & \text{if } ((i \leq k) \vee (j \leq k)) \wedge (i \neq j), \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

can be used to emphasize ranking at the top k elements. Misranking of pairs with at least one element ranked among the top k is penalized by this function. This can be of interest in applications such as information extraction or search engines where the ranking of the top documents matters more. For this emphasis function, all elements ranked below k are in a tie. In fact, it is possible to encode any tie relation using ω.

2.3.1 Bipartite ranking

In a bipartite ranking scenario, V is partitioned into a positive and negative set V^+ and V^- of sizes m^+ and m^- respectively, where $m^+ + m^- = |V| = n$. For this scenario (Balcan et al. 2008; Hanley and McNeil 1982; Lehmann 1975), we are often interested in the AUC score of $\sigma \in S(V)$ defined as follows:

$$1 - \mathrm{AUC}(V^+, V^-, \sigma) = \frac{1}{m^- m^+} \sum_{u \in V^+, v \in V^-} \sigma(v, u). \tag{2}$$

This expression measures the probability given a random *crucial* pair of elements, one of which is positive and the other negative, that the pair is misordered in σ. It is immediate to verify that this is equal to $L_\omega(\sigma, \sigma^*)$, where σ^* is any ranking placing V^+ ahead of V^-, and

$$\omega(i, j) = \frac{\binom{n}{2}}{m^- m^+} \begin{cases} 1 & (i \leq m^+) \wedge (j > m^+), \\ 1 & (j \leq m^+) \wedge (i > m^+), \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

2.3.2 Simplified notation

To avoid carrying σ^* and ω, we will define for convenience

$$\tau^*(u, v) = \sigma^*(u, v)\omega(\sigma^*(u), \sigma^*(v))$$

and

$$L(\sigma, \tau^*) := L_\omega(\sigma, \sigma^*) = \binom{n}{2}^{-1} \sum_{u \neq v} \sigma(u, v)\tau^*(v, u).$$

We will formally call τ^* a *generalized ranking*, and it will take the role of the ground truth. If ω is obtained as in (3) for some integers m^+, m^- satisfying $m^+ + m^- = n$ then we will say that the corresponding τ^* is *bipartite*.

It is immediate to verify from the properties of the weight function ω that for all $u, v, w \in V$,

$$\tau^*(u, v) \leq \tau^*(u, w) + \tau^*(w, v). \tag{4}$$

If τ^* is bipartite, then additionally,

$$\tau^*(u, v) + \tau^*(v, w) + \tau^*(w, u) = \tau^*(v, u) + \tau^*(w, v) + \tau^*(u, w). \tag{5}$$

2.4 Preference loss function

We need to extend the definition to measure the loss of a preference function h with respect to σ^*. In contrast with the loss function just defined, we need to define a *preference loss* measuring a generalized ranking's disagreements with respect to a preference function h when measured against τ^*. We can readily extend the loss definitions defined above as follows:

$$L(h, \tau^*) = L_\omega(h, \sigma^*) = \binom{n}{2}^{-1} \sum_{u \neq v} h(u, v)\tau^*(v, u).$$

As explained above, $L(h, \tau^*)$ is the ideal loss the learning algorithm will aim to achieve with the output ranking hypothesis σ.

Note that by our assumptions, h can take fractional values between zero and one. These can be interpreted as the preference function's confidence scores in the pairwise rankings. $L(h, \tau^*)$ can be viewed as the expected pairwise misranking loss with respect to the probability distribution defined by these confidence scores.

2.5 Input distribution

The set V we wish to rank together with the ground truth τ^* are drawn as a pair from a distribution we denote by D. In other words, τ^* may be a random function of V. For our analysis of the loss though, it is convenient to think of V and τ^* as fixed, because our bounds will be conditioned on fixed V, τ^* and will easily generalize to the stochastic setting. Finally, we say that D is bipartite if τ^* is bipartite with probability one.

2.6 Regret functions

The notion of regret is commonly used to measure the difference between the loss incurred by a learning algorithm and that of some *best* alternative. This section introduces the definitions of regret that we will be using to quantify the quality of a ranking algorithm in this context. We will define a notion of *weak* and *strong* regret for both ranking and classification losses as follows.

To define a strong ranking regret, we subtract from the loss function the minimal loss that could have been obtained from a global ranking $\tilde{\sigma}$ of U. More precisely, we define:

$$\mathcal{R}_{\text{rank}}(A, D) = E_{V,\tau^*,s}[L(A_s(V), \tau^*)] - \min_{\tilde{\sigma} \in S(U)} E_{V,\tau^*}[L(\tilde{\sigma}_{|V}, \tau^*)], \tag{6}$$

where $\tilde{\sigma}_{|V} \in S(V)$ is defined by restricting the ranking $\tilde{\sigma} \in S(U)$ to V in a natural way, and A is a possibly randomized algorithm using a stream of random bits s (and a pre-learned preference function h) to output a ranking $A_s(V)$ in $S(V)$.

As for the strong preference loss, it is natural to subtract the minimal loss over all, possibly cyclic, preference functions on U.

More precisely, we define:

$$\mathcal{R}_{\text{class}}(h, D) = E_{V,\tau^*}[L(h_{|V}, \tau^*)] - \min_{\tilde{h}} E_{V,\tau^*}[L(\tilde{h}_{|V}, \tau^*)], \tag{7}$$

where the minimum is over $\tilde{h}$, a preference function over U, and $\cdot_{|V}$ is a restriction operator on preference functions defined in the natural way.

The weak ranking and classification regret functions $\mathcal{R}'_{\text{rank}}$ and $\mathcal{R}'_{\text{class}}$ are defined as follows:

$$\mathcal{R}'_{\text{rank}}(A, D) = E_{V,\tau^*,s}[L(A_s(V), \tau^*)] - E_V \min_{\tilde{\sigma} \in S(V)} E_{\tau^*|V}[L(\tilde{\sigma}, \tau^*)], \tag{8}$$

$$\mathcal{R}'_{\text{class}}(h, D) = E_{V,\tau^*}[L(h_{|V}, \tau^*)] - E_V \min_{\tilde{h}} E_{\tau^*|V}[L(\tilde{h}, \tau^*)], \tag{9}$$

where $\tau^*|V$ is the random variable τ^* conditioned on fixed V. The difference between $\mathcal{R}$ and $\mathcal{R}'$ for both ranking and classification is that in their definition the min operator and the E_V operator are permuted.

The following inequalities follow from the concavity of min and Jensen's inequality:

$$\mathcal{R}'_{\text{rank}}(A, D) \geq \mathcal{R}_{\text{rank}}(A, D) \quad \text{and} \quad \mathcal{R}'_{\text{class}}(A, D) \geq \mathcal{R}_{\text{class}}(A, D). \tag{10}$$

For a fixed V and any $u, v \in V$, let

$$e(u, v) = E_{\tau^*|V}[\tau^*(u, v)]. \tag{11}$$

The reason we work with $\mathcal{R}'_{\text{class}}$ is because the preference function $\tilde{h}$ over U obtaining the min in the definition of $\mathcal{R}'_{\text{class}}$ can be determined locally for any $u, v \in U$ by

$$\tilde{h}(u, v) = \begin{cases} 1 & e(u, v) > e(v, u), \\ 0 & e(v, u) > e(u, v), \\ \mathbf{1}_{u>v} & \text{otherwise.} \end{cases} \tag{12}$$

Also, (4) holds with e replacing τ^*, and similarly for (5) if D is bipartite, by linearity of expectation. We cannot proceed in a similar way when working with the strong regret function $\mathcal{R}_{\text{class}}$.

The reason we work with weak ranking regret is for compatibility with our choice of weak classification regret, although our upper bounds on $\mathcal{R}'_{\text{rank}}$ trivially apply to $\mathcal{R}_{\text{rank}}$ in virtue of (10).

In Sect. 5.4, we will discuss certain assumptions under which our results work for the notion of strong regret as well.

2.7 A note on comparison with previous work

The original conference version of the paper by Balcan et al. (2007) appeared before our original conference version (Ailon and Mohri 2008), and their final journal version (Balcan

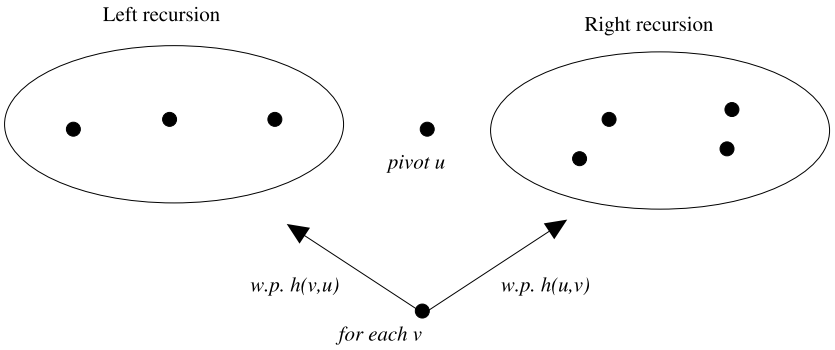

Fig. 1 QuickSort: unlike classical textbook assumptions, the comparison function we use is nondeterministic and may be nontransitive. The procedure still outputs a permutation, however

et al. 2008) appeared thereafter, with reference thereto. Balcan et al. claim in their Sect. 1 that our results on regret bounds (Theorem 2 here) could only *potentially* be comparable to the relevant results in Balcan et al. (2008) regarding AUC bounds. To the best of our knowledge, Theorem 2 here *is* indeed comparable with their result on AUC regret (modulo of course the important distinction that our result is based on a randomized algorithm and holds on expectation, while Balcan et al.'s (2008) is combinatorial). We also note here that Balcan et al. (2008) assume that the preference hypothesis (in our notation: h) depends on the set (in our notation: V). This renders their setting equivalent to our *weak regret* setting. To the confused reader, we recommend to assume that V is fixed throughout. Finally, Balcan et al. (2008) focus on the bipartite case, and then generalize their result in Sect. 6 to a *multi-partite* loss. From our standpoint, the multi-partite case considered by Balcan et al. (2008) is still only a strict special case of our general family of loss functions.

3 Algorithm for ranking using a preference function

This section describes and analyzes an algorithm for obtaining a global ranking of a subset using a prelearned preference function h, which corresponds to the second stage of the preference-based setting. Our bound on the loss will be derived using conditional expectation on the preference loss assuming a fixed subset $V \subseteq U$, and fixed ground truth τ^*.

3.1 Description

One simple idea to obtain a global ranking of the points in V consists of using a standard comparison-based sorting algorithm where the comparison operation is based on the preference function. However, since in general the preference function is not transitive, the property of the resulting permutation obtained is unclear.

This section shows however that the permutation generated by the standard QuickSort algorithm provides excellent guarantees.[2] Thus, the algorithm we suggest is the following (Fig. 1). Pick a random *pivot* element u uniformly at random from V. For each $v \neq u$, place

[2]We are not assuming here transitivity as in standard textbook presentations of QuickSort.

v on the left[3] of u with probability $h(v,u)$ and to its right with the remaining probability $h(u,v)$. Proceed recursively with the array to the left of u and the one to its right and return the concatenation of the permutation returned by the left recursion, u, and the permutation returned by the right recursion.

We will denote by $Q_s^h(V)$ the permutation obtained after running QuickSort on V using the preference function h, where s is the random stream of bits used by QuickSort for the selection of the pivots. The random stream s is also used to draw a random placement of u with respect to v when $h(u,v)$ is fractional—we do not describe this case in detail here. As we shall see in the next two sections, this algorithm produces high-quality global rankings in a time-efficient manner.

3.2 Ranking quality guarantees

The following theorems bound the ranking quality of the algorithm described, for both loss and regret, in the general and bipartite cases. The guarantees are given in expectation. We leave a detailed analysis of high probability bounds to future work.

Theorem 1 (Loss bounds in general case) *For any fixed subset $V \subseteq U$, preference function h on V, and generalized ranking τ^* on V, the following bound holds*:

$$\mathrm{E}_s\left[L\left(Q_s^h(V),\tau^*\right)\right] \leq 2L(h,\tau^*). \tag{13}$$

Taking the expectation of both sides, this implies immediately that

$$\mathrm{E}_{V,\tau^*,s}\left[L\left(Q_s^h(V),\tau^*\right)\right] \leq 2E_{V,\tau^*}[L(h,\tau^*)], \tag{14}$$

where h could depend on V.

Theorem 2 (Loss and regret bounds in bipartite case) *For any fixed $V \subseteq U$, preference function h over V, and bipartite generalized ranking τ^*, the following bound holds*:

$$\mathrm{E}_s\left[L\left(Q_s^h(V),\tau^*\right)\right] = L(h,\tau^*), \tag{15}$$

$$\mathcal{R}'_{\mathrm{rank}}\left(Q_s^h(\cdot),D\right) \leq \mathcal{R}'_{\mathrm{class}}(h,D). \tag{16}$$

Taking the expectation of both sides of (15), this implies immediately that if (V,τ^*) is drawn from a bipartite distribution D, then

$$\mathrm{E}_{V,\tau^*,s}\left[L\left(Q_s^h(V),\tau^*\right)\right] = E_{V,\tau^*}[L(h,\tau^*)], \tag{17}$$

where h can depend on V.

To present the proof of these theorems, we need some tools helpful in the analysis of QuickSort, similar to those originally developed by Ailon et al. (2005). The next section introduces these tools.

[3] We will use the convention that ranked items are written from left to right, starting with the most preferred ones.

3.3 Analysis of QuickSort

Assume V is fixed, and let $Q_s = Q_s^h(V)$ be the (random) ranking output by QuickSort on V using the preference function h. During the execution of QuickSort, the order between two elements $u, v \in V$ is determined in one of two ways:

- Directly: u (or v) was selected as the pivot with v (resp. u) present in the same sub-array in a recursive call to QuickSort. We denote by $p_{uv} = p_{vu}$ the probability of that event. Conditioned on this case, the algorithm orders u and v by flipping a biased coin according to the preference function h.
- Indirectly: a third element $w \in V$ is selected as pivot with w, u, v all present in the same sub-array in a recursive call to QuickSort, u is assigned to the left sub-array and v to the right (or vice-versa).

  Let p_{uvw} denote the probability of the event that u, v, and w are present in the same array in a recursive call to QuickSort and that one of them is selected as pivot. Note that conditioned on that event, each of these three elements is equally likely to be selected as a pivot since the pivot selection is based on a uniform distribution.

  If (say) w is selected among the three, then u will be placed on the left of v with probability $h(u, w)h(w, v)$ and to its right with probability $h(v, w)h(w, u)$. In all other cases, the order between u, v will be determined only in a deeper nested call to QuickSort.

Let $X, Y\colon V \times V \to \mathbb{R}$ be any two functions on ordered pairs $u, v \in V$, and let $Z\colon \binom{V}{2} \to \mathbb{R}$ be a function on unordered pairs. We define three functions $\alpha[X, Y]\colon \binom{V}{2} \to \mathbb{R}$, $\beta[X]\colon \binom{V}{3} \to \mathbb{R}$ and $\gamma[Z]\colon \binom{V}{3} \to \mathbb{R}$ as follows:

$$\begin{aligned}
\alpha[X, Y]_{uv} &= X(u, v)Y(v, u) + X(v, u)Y(u, v),\\
\beta[X]_{uvw} &= \frac{1}{3}(h(u, v)h(v, w)X(w, u) + h(w, v)h(v, u)X(u, w))\\
&\quad + \frac{1}{3}(h(v, u)h(u, w)X(w, v) + h(w, u)h(u, v)X(v, w))\\
&\quad + \frac{1}{3}(h(u, w)h(w, v)X(v, u) + h(v, w)h(w, u)X(u, v)),\\
\gamma[Z]_{uvw} &= \frac{1}{3}(h(u, v)h(v, w) + h(w, v)h(v, u))Z_{uw}\\
&\quad + \frac{1}{3}(h(v, u)h(u, w) + h(w, u)h(u, v))Z_{vw}\\
&\quad + \frac{1}{3}(h(u, w)h(w, v) + h(v, w)h(w, u))Z_{uv}.
\end{aligned}$$

Lemma 1 (QuickSort decomposition)

1. *For any* $Z\colon \binom{V}{2} \to \mathbb{R}$,

$$\sum_{u<v} Z_{uv} = \sum_{u<v} p_{uv} Z_{uv} + \sum_{u<v<w} p_{uvw}\gamma[Z]_{uvw}.$$

2. *For any* $X\colon V \times V \to \mathbb{R}$,

$$E_s\left[\sum_{u<v} \alpha[Q_s, X]_{uv}\right] = \sum_{u<v} p_{uv}\alpha[h, X]_{uv} + \sum_{u<v<w} p_{uvw}\beta[X]_{uvw}.$$

Recall that we allow the notation $u < v$ under the summation symbols in the lemma by assuming that the set V is arbitrarily ordered.

Proof To see the first part, notice that for every unordered pair $u < v$ the expression Z_{uv} is accounted for on the RHS of the equation with total coefficient:

$$p_{uv} + \sum_{w \notin \{u,v\}} \frac{1}{3} p_{uvw}(h(u,w)h(w,v) + h(v,w)h(w,u)).$$

Now, p_{uv} is the probability that the order of (u, v) is determined directly (by definition), and

$$\frac{1}{3} p_{uvw}(h(u,w)h(w,v) + h(v,w)h(w,u))$$

is the probability that their order is determined indirectly via w as pivot. Since each pair's ordering is accounted for exactly once, these probabilities are for pairwise disjoint events that cover the probability space. Thus, the total coefficient of Z_{uv} on the RHS is one, as is on the LHS. The second part is proved similarly. □

3.4 Loss bounds

This section proves Theorem 1 and the first part of Theorem 2. For a fixed τ^*, the loss incurred by QuickSort is $L(Q_s, \tau^*) = \binom{n}{2}^{-1} \sum_{u<v} \alpha[Q_s, \tau^*]_{uv}$. By the second part of Lemma 1, the expected loss is therefore

$$\mathrm{E}_s[L(Q_s, \tau^*)] = \binom{n}{2}^{-1} \left(\sum_{u<v} p_{uv} \alpha[h, \tau^*]_{uv} + \sum_{u<v<w} p_{uvw} \beta[\tau^*]_{uvw} \right). \tag{18}$$

Also, the following holds by definition of L:

$$L(h, \tau^*) = \binom{n}{2}^{-1} \sum_{u<v} \alpha[h, \tau^*]_{uv}. \tag{19}$$

Thus, by the first part of Lemma 1,

$$L(h, \tau^*) = \binom{n}{2}^{-1} \left(\sum_{u<v} p_{uv} \alpha[h, \tau^*]_{uv} + \sum_{u<v<w} p_{uvw} \gamma[\alpha[h, \tau^*]]_{uvw} \right). \tag{20}$$

To complete the proof, it suffices to show that for all u, v, w,

$$\beta[\tau^*]_{uvw} \leq 2\gamma[\alpha[h, \tau^*]]_{uvw}, \tag{21}$$

and that if τ^* is bipartite, then

$$\beta[\tau^*]_{uvw} = \gamma[\alpha[h, \tau^*]]_{uvw}. \tag{22}$$

Both functions $\beta[\tau*]_{uvw}$ and $\gamma[\alpha[h, \tau^*]]_{uvw}$ are linear in the six variables $\tau^*(a, b)$ for $\{a, b\} \subseteq \{u, v, w\}$. Therefore, it suffices to check only the vertices of the polytope defining these variables. Up to symmetries, there are three vertices $V1$, $V2$ and $V3$ to consider. The vertex $V1$ corresponds to a complete ordering (a permutation) of u, v, w. The vertex $V2$ corresponds to a tie between two of u, v, w and some ordering between these tied elements and the third, untied element. The vertex $V3$ corresponds to a tie among all of u, v, w.

(V1) $(\tau^*(u,v),\tau^*(v,u),\tau^*(v,w),\tau^*(w,v),\tau^*(w,u),\tau^*(u,w)) = (0,1,0,1,1,0)$;
(V2) $(\tau^*(u,v),\tau^*(v,u),\tau^*(v,w),\tau^*(w,v),\tau^*(w,u),\tau^*(u,w)) = (1,0,0,0,0,1)$;
(V3) $(\tau^*(u,v),\tau^*(v,u),\tau^*(v,w),\tau^*(w,v),\tau^*(w,u),\tau^*(u,w)) = (0,0,0,0,0,0)$.

Vertex (V1) corresponds to a ground truth τ^* preferring w to v and v to u. Vertex (V2) correspond to a bipartite ground truth τ^* which prefers u to v and w, but ties together v and w. Vertex (V3) corresponds to a ground truth τ^* tying all of u, v, w together. Clearly for (V3) $\beta[\tau^*]_{uvw} = \gamma[\alpha[h,\tau^*]]_{uvw} = 0$ and we are done.

To complete the proof of Theorems 1 and 2, it suffices to show that in case (V1) $\beta[\tau^*]_{uvw} \leq 2\gamma[\alpha[h,\tau^*]]_{uvw}$ and that in case (V2) $\beta[\tau^*]_{uvw} \leq \gamma[\alpha[h,\tau^*]]_{uvw}$.

Case (V1): Plugging in the value of the τ^*-variables in the definition of β and γ yields

$$\beta[\tau^*]_{uvw} = \frac{1}{3}(h(u,v)h(v,w) + h(v,u)h(u,w) + h(w,v)h(u,w)),$$

$$\gamma[\alpha[h,\tau^*]]_{uvw} = \frac{1}{3}(1 + (-1 + 2h(u,v)h(v,w))h(w,u)).$$

Let $f = \beta[\tau^*]_{uvw} - 2\gamma[\alpha[h,\tau^*]]_{uvw}$. To simplify notation, we let $h_1 = h(u,v) = 1 - h(v,u)$, $h_2 = h(v,w) = 1 - h(w,v)$, $h_3 = h(w,u) = 1 - h(u,w)$. This gives $f = \frac{1}{3}(h_2(h_3 - 1) + h1(-1 + h_2 + h_3 - 4h_2h_3))$. We need to prove that $\sup f \leq 0$ where the supremum is taken on the set $h_1, h_2, h_3 \in [0,1]$. To do this it suffices to check that $\sup\{-1 + h_2 + h_3 - 4h_2h_3\}$ is non-positive. Using simple calculus tools we can check that this supremum is $-3/4$, as required.

Case (V2): Plugging in the value of the τ^*-variables in the definition of $\beta[\Delta]_{uvw}$ and $\gamma[\alpha[h,\Delta]]_{uvw}$, it is easy to verify that $\beta[\Delta]_{uvw} = \gamma[\alpha[h,\Delta]]_{uvw}$, giving the required result.

We now examine a consequence of Theorem 1 for QuickSort that can be compared with the bound given by Cohen et al. (1999) for a greedy algorithm based on the tournament degree. Let σ_{optimal} be the ranking with the least amount of pairwise disagreement with h:

$$\sigma_{\text{optimal}} = \arg\min_{\sigma} L(h,\sigma).$$

Then, the following corollary bounds the expected pairwise disagreement of QuickSort with respect to σ_{optimal} by twice that of the preference function with respect to σ_{optimal}.

Corollary 1 *For any $V \subseteq U$ and preference function h over V, the following bound holds*:

$$\mathrm{E}_s\big[L\big(Q_s^h(V),\sigma_{\text{optimal}}\big)\big] \leq 2L(h,\sigma_{\text{optimal}}). \tag{23}$$

The corollary is immediate since technically any ranking, in particular σ_{optimal}, can be taken as σ^* in the proof of Theorem 1.

Corollary 2 *Let $V \subseteq U$ be an arbitrary subset of U and let σ_{optimal} be as above. Then, the following bound holds for the pairwise disagreement of the ranking $Q_s^h(V)$ with respect to h*:

$$\mathrm{E}_s\big[L\big(h, Q_s^h(V)\big)\big] \leq 3L(h,\sigma_{\text{optimal}}). \tag{24}$$

Proof The result follows directly Corollary 1 and the application of the triangle inequality. □

This result is in fact known from previous work (Ailon et al. 2005; Ailon 2007) where it is proven directly without resorting to the intermediate inequality (23). In fact, a better factor of 2.5 is known to be achievable from that work using a more complicated algorithm, which gives hope for a 1.5 bound improving Theorem 1.

3.5 Regret bounds for bipartite case

This section proves the second part of Theorem 2, that is the regret bound. Since in the definition of $\mathcal{R}'_{\text{rank}}$ and $\mathcal{R}'_{\text{class}}$ the expectation over V is outside the min operator, we may continue to fix V. Let D_V denote the distribution over the bipartite τ^* conditioned on V. By the definitions of $\mathcal{R}'_{\text{rank}}$ and $\mathcal{R}'_{\text{class}}$, it is now sufficient to prove that

$$\mathrm{E}_{\tau^*|V,s}\big[L\big(Q_s^h,\tau^*\big)\big]-\min_{\tilde{\sigma}}\mathrm{E}_{\tau^*|V}[L(\tilde{\sigma},\tau^*)]\leq \mathrm{E}_{\tau^*|V}[L(h,\tau^*)]-\min_{\tilde{h}}\mathrm{E}_{\tau^*|V}[L(\tilde{h},\tau^*)]. \quad (25)$$

We let $e(u,v)$ denote $\mathrm{E}_{\tau^*|V}[\tau^*(u,v)]$, then by the linearity of expectation, it follows that $\mathrm{E}_{\tau^*|V}[L(\tilde{\sigma},\tau^*)]=L(\tilde{\sigma},e)$ and similarly $\mathrm{E}_{\tau^*|V}[L(\tilde{h},\tau^*)]=L(\tilde{h},e)$. Thus, inequality 25 can be rewritten as

$$\mathrm{E}_s\big[L\big(Q_s^h,e\big)\big]-\min_{\tilde{\sigma}}L(\tilde{\sigma},e)\leq L(h,e)-\min_{\tilde{h}}L(\tilde{h},e). \quad (26)$$

Now let $\tilde{\sigma}$ and $\tilde{h}$ be the minimizers of the min operators on the left and right sides, respectively. Recall that for all $u,v\in V$, $\tilde{h}(u,v)$ can be taken greedily as a function of $e(u,v)$ and $e(v,u)$, as in (12):

$$\tilde{h}(u,v)=\begin{cases}1 & e(u,v)>e(v,u),\\ 0 & e(u,v)<e(v,u),\\ \mathbf{1}_{u>v} & \text{otherwise (equality).}\end{cases} \quad (27)$$

Using Lemma 1 and linearity, the LHS of (26) can be rewritten as:

$$\binom{n}{2}^{-1}\Bigg(\sum_{u<v}p_{uv}\alpha[h-\tilde{\sigma},e]_{uv}+\sum_{u<v<w}p_{uvw}(\beta[e]-\gamma[\alpha[\tilde{\sigma},e]])_{uvw}\Bigg),$$

and the RHS of (26) as:

$$\binom{n}{2}^{-1}\Bigg(\sum_{u<v}p_{uv}\alpha[h-\tilde{h},e]_{uv}+\sum_{u<v<w}p_{uvw}\gamma[\alpha[h-\tilde{h},e]]_{uvw}\Bigg).$$

Now, clearly, for all (u,v) by construction of $\tilde{h}$, we must have $\alpha[h-\tilde{\sigma},e]_{uv}\leq\alpha[h-\tilde{h},e]_{uv}$. To conclude the proof of the theorem, we define $F\colon\binom{n}{3}\to\mathbb{R}$ as follows:

$$F=\beta[e]-\gamma[\alpha[\tilde{\sigma},e]]-(\gamma[\alpha[h,e]]-\gamma[\alpha[\tilde{h},e]]). \quad (28)$$

It now suffices to prove that $F_{uvw}\leq 0$ for all $u,v,w\in V$. Clearly F is a function of the values of

$$e(a,b) : \{a,b\} \subseteq \{u,v,w\}, \tag{29}$$

$$h(a,b) : \{a,b\} \subseteq \{u,v,w\}, \tag{30}$$

$$\tilde{\sigma}(a,b) : \{a,b\} \subseteq \{u,v,w\}. \tag{31}$$

Recall that $\tilde{h}$ depends on e. By (4) and (5), the e-variables can take values satisfying the following constraints for all $u, v, w \in V$:

$$\forall \{a,b,c\} = \{u,v,w\},\ e(a,c) \le e(a,b) + e(b,c), \tag{32}$$

$$e(u,v) + e(v,w) + e(w,u) = e(v,u) + e(w,v) + e(u,w), \tag{33}$$

$$\forall a,b \in \{u,v,w\}, \quad e(a,b) \ge 0. \tag{34}$$

Let $P \subseteq \mathbb{R}^6$ denote the polytope defined by (32)–(34) in the variables $e(a,b)$ for $\{a,b\} \subseteq \{u,v,w\}$. We subdivide P into smaller subpolytopes on which the $\tilde{h}$ variables are constant. Up to symmetries, we can consider only two cases: (i) $\tilde{h}$ induces a cycle on u, v, w and (ii) $\tilde{h}$ is cycle-free on u, v, w.

(i) Without loss of generality, assume $\tilde{h}(u,v) = \tilde{h}(v,w) = \tilde{h}(w,u) = 1$. But this implies that $e(u,v) \ge e(v,u)$, $e(v,w) \ge e(w,v)$ and $e(w,u) \ge e(u,w)$. Together with (33) and (34), this implies that $e(u,v) = e(v,u)$, $e(v,w) = e(w,v)$, and $e(w,u) = e(u,w)$. Consequently,

$$\beta[e]_{uvw} = \gamma[\alpha[\tilde{\sigma}, e]]_{uvw} \tag{35}$$

$$= \gamma[\alpha[h, e]]_{uvw} = \gamma[\alpha[\tilde{h}, e]]_{uvw} \tag{36}$$

$$= \frac{1}{3}(e(u,v) + e(v,w) + e(w,u)), \tag{37}$$

and $F_{uvw} = 0$, as required.

(ii) Without loss of generality, assume $\tilde{h}(u,v) = \tilde{h}(v,w) = \tilde{h}(u,w) = 1$. This implies that

$$e(u,v) \ge e(v,u), \tag{38}$$

$$e(v,w) \ge e(w,v), \tag{39}$$

$$e(u,w) \ge e(w,u). \tag{40}$$

Let $\tilde{P} \subseteq P$ denote the polytope defined by (38) and (32)–(34). Clearly, F is linear in the six e variables when all the other variables are fixed. Since F is also homogeneous in the e variables, it suffices to prove that $F \le 0$ for e taking values in $\tilde{P}' \subseteq \tilde{P}$, which is defined by adding the constraint, say,

$$\sum_{a,b \in \{u,v,w\}} e(a,b) = 2.$$

It is now enough to prove that $F \le 0$ for e being a vertex of $\tilde{P}'$. This finite set of cases can be easily checked to be:

$$(e(u,v), e(v,u), e(u,w), e(w,u), e(w,v), e(v,w)) \in A \cup B,$$

where

$$A = \{(0,0,1,0,0,1),(1,0,1,0,0,0)\},$$
$$B = \{(.5,.5,.5,.5,0,0),(.5,.5,0,0,.5,.5),(0,0,.5,.5,.5,.5)\}.$$

The points in B were already checked in case (i), which is, geometrically, a boundary of case (ii). It remains to check the two points in A.

– case $(0,0,1,0,0,1)$: plugging in the definitions, one checks that:

$$\begin{aligned}\beta[e]_{uvw} &= \frac{1}{3}(h(w,v)h(v,u) + h(w,u)h(u,v)),\\ \gamma[\alpha[h,e]]_{uvw} &= \frac{1}{3}((h(u,v)h(v,w) + h(w,v)h(v,u))h(w,u)\\ &\quad + (h(v,u)h(u,w) + h(w,u)h(u,v))h(w,v)),\\ \gamma[\alpha[\tilde{h},e]]_{uvw} &= 0.\end{aligned}$$

But, from the fact that $h(w,u) + h(u,w) = 1$ and $h(w,v) + h(v,w) = 1$, it is clear that $\beta[e]_{uvw} \equiv \gamma[\alpha[h,e]]_{uvw}$, which implies $F \equiv -\gamma[\alpha[\tilde{\sigma},e]] \leq 0$.

– case $(1,0,1,0,0,0)$: plugging in the definitions, one can check that:

$$\begin{aligned}\beta[e]_{uvw} &= \frac{1}{3}(h(w,v)h(v,u) + h(v,w)h(w,u)),\\ \gamma[\alpha[h,e]]_{uvw} &= \frac{1}{3}((h(u,v)h(v,w) + h(w,v)h(v,u))h(w,u)\\ &\quad + (h(u,w)h(w,v) + h(v,w)h(w,u))h(v,u)),\\ \gamma[\alpha[\tilde{h},e]]_{uvw} &= 0.\end{aligned}$$

Again, as in case (i), one can check that $\beta[e]_{uvw} \equiv \gamma[\alpha[h,e]]_{uvw}$ and consequently $F \leq 0$ again.

This concludes the proof of the second part of Theorem 2.

3.6 Time complexity

In this section we show that running QuickSort does not require $\Omega(n^2)$ accesses to $h_{u,v}$. In what follows, we use the term *tournament* to denote a set V equipped with a preference function h on V.

Theorem 3 *The expected number of calls to the preference function made by QuickSort for a tournament of size n is at most $O(n \log n)$. Moreover, if only the top k elements are sought, then the bound is reduced to $O(k \log k + n)$ by pruning the recursion.*

It is well known that QuickSort on cycle-free tournaments runs in time $O(n \log n)$, where n is the size of the set we wish to sort. That this holds for QuickSort on general tournaments is a bit more difficult. We will need the following lemma.

Lemma 2 *Fix a tournament $G = (V, h)$ of size n. Let G_L, G_R denote the left and right random sub-tournaments induced after a pivoting step in QuickSort. Let n_L and n_R denote their sizes, respectively. Then* $\mathrm{E}[n_L] = \mathrm{E}[n_R] = (n-1)/2$ *and* $\mathrm{E}[n_L^2], \mathrm{E}[n_R^2] \leq n^2/2.9$ *for* $n > 100$.

Proof The main observation for the expectation claim is that each vertex $v \in V$ is assigned to the left recursion with probability exactly $\mathrm{outdeg}(v)/(n-1)$ and to the right with probability $\mathrm{indeg}(v)/(n-1)$, over the choice of the pivot, where $\mathrm{outdeg}(v) = \sum_{u \neq v} h(v,u)$ and $\mathrm{indeg}(v) = \sum_{u \neq v} h(u,v)$. Therefore, the expected size of both the left and right recursions is exactly $(n-1)/2$, proving the claim for the expectation. As for the second moment,

$$\mathrm{E}[n_R^2] = \frac{1}{n} \sum_u \left(\sum_v h(u,v) + 2 \sum_{v<w} h(u,v)h(u,w) \right) = \frac{n-1}{2} + \frac{2}{n} \sum_{u<v<w} H_{uvw},$$

where the variables u, v, w take distinct values in the nested summation and where H_{uvw} is a shorthand for

$$h(u,v)h(u,w) + h(v,u)h(v,w) + h(w,u)h(w,v). \tag{41}$$

It is straightforward to check that $\max\{H_{uvw} : h \text{ preference function}\} = 1$. The maximum can be reached for say $h(u,v) = h(u,w) = 1$. Thus, $\mathrm{E}[n_R^2] \leq n/2 + n^2/3 \leq n^2/2.9$ for $n > 100$. □

We are now ready to prove Theorem 3.

Proof Let the random variable $T(G)$ denote the expected running time of QuickSort on a tournament $G = (V, h)$. By Lemma 2 and Markov's inequality, it follows that for $n > 100$

$$\Pr[n_R > 9n/10] \leq 0.43. \tag{42}$$

And similarly for n_L. Hence by the union bound, the following holds:

$$\Pr[\max\{n_L, n_R\} > 9n/10] \leq 0.9. \tag{43}$$

We can assume that QuickSort always chooses to first recurse on a subtournament of size more than $9n/10$ if such a (unique) subtournament exists: in other words, it first tries to break down the input into chunks of size at most $9n/10$. The expected number of times it may take to reach this point, in virtue of (43) is at most a constant $c_1 > 0$ (expectation of a geometric random variable) and the expected total number of comparisons it takes to reach this point is therefore at most $c_1 n$. Assume that at this point we have q subtournaments $G_1, \ldots, G_q$ to recurse on, and their sizes are $n_1, n_2, \ldots, n_q$ (with $n_1 + \cdots + n_q = n$ and $n_i \leq 9n/10$ by assumption). By an inductive argument which we sketch, the base case being $n \leq 100$, for a sufficiently large constant $c_2 > 0$, the total expected running time is bounded as follows:

$$\begin{aligned} T(G) &\leq c_1 n + \sum T(G_i) \\ &\leq c_1 n + \sum c_2 n_i \log(n_i) \\ &\leq c_1 n + \sum c_2 n_i \log(9n/10) \end{aligned}$$

$$= c_1 n + c_2 n \log(9n/10)$$
$$= c_1 n + c_2 n \log n + c_2 n \log(9/10)$$
$$\leq c_2 n \log n.$$

We now prove the second part of the theorem. Assume that only the k first elements of the output are sought, that is, we are interested in outputting only elements in positions $1, \ldots, k$. The algorithm which we denote by k-QuickSort is clear: recurse with $\min\{k, n_L\}$-QuickSort on the left side and $\max\{0, k - n_L - 1\}$-QuickSort on the right side, where n_L is the size of the left recursion, and 0-QuickSort takes zero steps by assumption. To make the analysis simpler, we will assume that whenever $k \geq n/10$, k-QuickSort simply returns the output of the standard QuickSort, which runs in expected time $O(n \log n) = O(n + k \log k)$, within the sought bound. Let $t_k(G)$ denote the running time of k-QuickSort on a tournament G, where $k \leq n/10$ and the size of G is n. We get:

$$t_k(G) = n - 1 + t_{\min\{k,n_L\}}(G_L) + t_{\max\{0,k-n_L-1\}}(G_R). \tag{44}$$

Assume by induction that for all $\{k', n' \colon k' \leq n' < n\}$ and for all tournaments G' on n' vertices,

$$\mathrm{E}[t_{k'}(G')] \leq cn' + c'k' \log k'$$

for some global $c, c' > 0$. This is clearly true for some c, c' for $n \leq 100$. Then, by conditioning on G_L, G_R, taking expectations on both sides of (44) and by induction,

$$\mathrm{E}[t_k(G) \mid G_L, G_R] \leq n - 1 + cn_L + c' \min\{k, n_L\} \log \min\{k, n_L\} + cn_R \mathbf{1}_{n_L < k-1} + c' \max\{k - n_L - 1, 0\} \log \max\{k - n_L - 1, 0\}.$$

By convexity and monotonicity of the function $x \mapsto x \log x$,

$$\min\{k, n_L\} \log \min\{k, n_L\} + \max\{k - n_L - 1, 0\} \log \max\{k - n_L - 1, 0\} \leq k \log k.$$

Thus,

$$\mathrm{E}[t_k(G) \mid G_L, G_R] \leq n - 1 + cn_L + cn_R \mathbf{1}_{n_L < k-1} + c'k \log k. \tag{45}$$

By conditional expectation,

$$\mathrm{E}[t_k(G)] \leq n - 1 + c(n-1)/2 + c'k \log k + c\mathrm{E}[n_R \mathbf{1}_{n_L < k-1}].$$

To complete the inductive hypothesis, we need to bound $\mathrm{E}[n_R \mathbf{1}_{n_L < k-1}]$, which is bounded by $n \Pr[n_L < k - 1]$. The event $\{n_L < k - 1\}$ is equivalently written as $\{n_R > n - k\}$. Since $n - k \geq 9n/10$, we can use the bound (42) to conclude that

$$\mathrm{E}[t_k(G)] \leq n - 1 + c(n-1)/2 + c'k \log k + 0.43 \times cn,$$

which is at most $cn + c'k \log k$ for sufficiently large c, as required. □

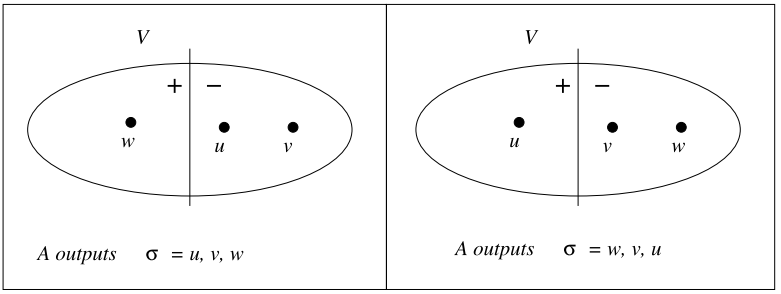

Fig. 2 Response of adversary to algorithm A in lower bound proof

4 Lower bounds

Let r denote the classification regret. Balcan et al. (2007) proved a lower bound of $2r$ for the regret of the algorithm MFAT defined as the solution to the minimum feedback arc-set problem on the tournament V with an edge (u, v) when $h(u, v) = 1$. More precisely, they showed an example of fixed V, h, and bipartite generalized ranking τ^* on V, such that the classification regret of h tends to $1/2$ of the ranking regret of MFAT on V, h. Note that in this case, since τ^* is a fixed function of V, the regret and loss coincide both for classification and for ranking.

Here we give a simple proof of a more general theorem stating that same bound holds for *any* deterministic algorithm, including of course MFAT.

Theorem 4 *For any* deterministic algorithm A *taking as input* $V \subseteq U$ *and a binary preference function* h *on* V *and outputting a ranking* $\sigma \in S(V)$, *there exists a bipartite distribution* D *on* (V, τ^*) *such that*

$$\mathcal{R}_{\text{rank}}(A, D) \geq 2\mathcal{R}_{\text{class}}(h, D). \tag{46}$$

Note that the theorem implies that, in the bipartite case, no deterministic algorithm converting a preference function into a linear ranking can do better than our randomized algorithm (on expectation). Thus, randomization is essentially necessary in this setting.

The proof is based on an adversarial argument. In our construction, the support of D is reduced to a single pair (V, τ^*) (deterministic input), thus the loss and both the weak and strong regrets coincide and a similar argument applies to the loss function and the weak regret functions.

Proof Fix $V = \{u, v, w\}$, and let the support of D be reduced to (V, τ^*), where the bipartite generalized ranking τ^* is one that we will select adversarially. Assume a cycle: $h(u, v) = h(v, w) = h(w, u) = 1$. Up to symmetry, there are two options for the output σ of A on V, h:

1. $\sigma(u) < \sigma(v) < \sigma(w)$: in this case, the adversary can choose τ^* corresponding to the partition $V^+ = \{w\}$ and $V^- = \{u, v\}$. Clearly, $\mathcal{R}_{\text{class}}(h, D)$ now equals $1/2$ since h is penalized only for misranking the pair (v, w), but $\mathcal{R}_{\text{rank}}(A, D) = 1$ since σ is misordering both (u, w) and (v, w).
2. $\sigma(w) < \sigma(v) < \sigma(u)$: in this case, the adversary can choose τ^* corresponding to the partition $V^+ = \{u\}$ and $V^- = \{v, w\}$. Similarly, $\mathcal{R}_{\text{class}}(h, D)$ now equals $1/2$ since h is

penalized only for misranking the pair (u, w), while $\mathcal{R}_{\text{rank}}(A, D) = 1$ since σ is misordering both (u, v) and (u, w). □

5 Discussion

5.1 History of QuickSort

The textbook algorithm, by now standard, was originally discovered by Hoare (1961). Montague and Aslam (2002) experimented with QuickSort for information retrieval (IR) by aggregating rankings from different sources of retrieval. They claimed an $O(n \log n)$ time bound on the number of comparisons, although the proof seemed to rely on the folklore QuickSort proof without addressing the non-transitivity problem. They proved certain combinatorial bounds on the output of QuickSort and provided an empirical justification of its IR merits. Ailon et al. (2005) also considered the rank aggregation problem and proved theoretical cost bounds for many ranking problems on weighted tournaments. They strengthened these bounds by considering non-deterministic pivoting rules arising from solutions to certain ranking LP's. This work was later extended by Ailon (2007) to deal with rankings with ties, in particular, top-k rankings. Hedge et al. (2007) and Williamson and van Zuylen (2007) derandomized the random pivot selection step in QuickSort for many of the combinatorial optimization problems studied by Ailon et al. (2005).

5.2 The decomposition technique

The technique developed in Lemma 1 is very general and can be used for a wide variety of loss functions and variants of QuickSort involving non-deterministic ordering rules (Ailon et al. 2005; Ailon 2007). Such results would typically amount to bounding $\beta[X]_{uvw}/\gamma[Z]_{uvw}$ for some carefully chosen functions X, Z depending on the application.

5.3 Combinatorial optimization vs. learning of ranking

QuickSort, sometimes referred to as FAS-Pivot in that context, was used by Ailon et al. (2005) and Ailon (2007) to approximate certain NP-Hard weighted instances of the problem of minimum feedback arcset in tournaments (Alon 2006). There is much similarity between the techniques used in that work and those of the analyses of this work, but there is also a significant difference that should be noted.

In the minimum feedback arc-set problem, we are given a tournament G and wish to find an acyclic tournament H on the same vertex set minimizing $\Delta(G, H)$, where Δ counts the number of edges pointing in opposite directions between G, H (or a weighted version thereof). However, the cost we are considering is $\Delta(G, H_\sigma)$ for some fixed acyclic tournament H_σ induced by some permutation σ (the ground truth). In this work, we showed in fact that if G' is obtained from G using QuickSort, then $\mathrm{E}[\Delta(G', H_\sigma)] \leq 2\Delta(G, H_\sigma)$ for *any* σ (Theorem 1). If H is the optimal solution to the (weighted) minimum feedback arc-set problem corresponding to G, then it is easy to see that $\Delta(H, H_\sigma) \leq \Delta(G, H) + \Delta(G, H_\sigma) \leq 2\Delta(G, H_\sigma)$. However, recovering H is NP-Hard in general. Approximating $\Delta(G, H)$ modulo a constant factor $1 + \varepsilon$ using an acyclic tournament H', as in the combinatorial optimization world, only guarantees a constant factor of $2 + \varepsilon$:

$$\Delta(H', H_\sigma) \leq \Delta(G, H') + \Delta(G, H_\sigma) \leq (1+\varepsilon)\Delta(G, H) + \Delta(G, H_\sigma)$$
$$\leq (2+\varepsilon)\Delta(G, H_\sigma).$$

Thus, this work also adds a significant contribution to Ailon et al. (2005), Ailon (2007) and Kenyon-Mathieu and Schudy (2007).

5.4 Weak vs. strong regret functions

For the proof of the regret bound of Theorem 2 we used the fact that the minimizer $\tilde{h}$ in the definition (8)–(9) of $\mathcal{R}'_{\text{class}}$ could be determined independently for each pair $u, v \in U$, using (12). This could also be done for strong regrets if the distribution D on V, τ^* satisfied the following pairwise IIA condition.

Definition 1 A distribution D on subsets $V \subseteq U$ and generalized rankings τ^* on V satisfies the *pairwise independence on irrelevant alternatives* (pairwise IIA) if for all $u, v \in U$ and for any two subsets $V_1, V_2 \supseteq \{u, v\}$,

$$E_{\tau^*|V_1}[\tau^*(u, v)] = E_{\tau^*|V_2}[\tau^*(u, v)].$$

Note: we chose the terminology IIA in agreement with its use in Arrow's seminal work (Arrow 1950) for a similar notion.

When pairwise IIA holds, the average ground truth relation between u and v, conditioned on u, v included in V, is independent of V.

Recall that a bipartite τ^* is derived from a pair σ^*, ω, where ω is defined using a term $1/m^- m^+$, for compatibility with the definition of AUC. The numbers m^+ and m^- depend on the underlying size of the positive and negative sets partitioning of V and therefore cannot be inferred from (u, v) alone. Thus, in the standard bipartite case, the pairwise IIA assumption is not natural. If, however, we replaced our definitions in the bipartite case and used the following:

$$\omega(i, j) = \begin{cases} 1 & (i \leq m^+) \wedge (j > m^+), \\ 1 & (j \leq m^+) \wedge (i > m^+), \\ 0 & \text{otherwise}, \end{cases} \tag{47}$$

instead of (3), then it would be reasonable to believe that pairwise IIA does hold in the bipartite case. In fact, it would be reasonable to make the stronger assumption that for any fixed $u, v \in U$ and $V_1, V_2 \supseteq \{u, v\}$, the distribution of the random variables $\tau^*(u, v)|V_1$ and $\tau^*(u, v)|V_2$ are equal. This corresponds to the intuition that when comparing a pair (u, v) in a context of a set V containing them, human labelers are not as influenced by the irrelevant information $V \backslash \{u, v\}$ as they would be by $V \backslash \{u\}$ when asked to evaluate single elements u. The irrelevant information in V is often referred to as *anchor* in experimental psychology and economics (Ariely et al. 2008).

Our regret bounds would still hold if we used (47), but we chose (3) to present our results in terms of the familiar average pairwise misranking error or AUC loss.

Another possible assumption allowing usage of strong regrets is to let the preference function learned in the first stage depend on V. This is, in fact, assumption made by Balcan et al. (2008).

6 Conclusion

We described a reduction of the learning problem of ranking to classification. The efficiency of this reduction makes it practical for large-scale information extraction and search engine applications. A finer analysis of QuickSort is likely to further improve our reduction bound by providing a concentration inequality for the algorithm's deviation from its expected behavior using the confidence scores output by the classifier. Our reduction leads to a competitive ranking algorithm that can be viewed as an alternative to the algorithms previously designed for the score-based setting.

Acknowledgements We thank Alina Beygelzimer and John Langford for helpful discussions. Mehryar Mohri's work was partially funded by the New York State Office of Science Technology and Academic Research (NYSTAR).

References

Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S., & Roth, D. (2005). Generalization bounds for the area under the roc curve. *Journal of Machine Learning Research*, *6*, 393–425.

Agarwal, S., & Niyogi, P. (2005). Stability and generalization of bipartite ranking algorithms. In *COLT* (pp. 32–47).

Ailon, N. (2007). Aggregation of partial rankings, p-ratings and top-m lists. In *SODA*.

Ailon, N., Charikar, M., & Newman, A. (2005). Aggregating inconsistent information: ranking and clustering. In *Proceedings of the 37th annual ACM symposium on theory of computing* (pp. 684–693). Baltimore, MD, USA, May 22–24, 2005. New York: ACM.

Ailon, N., & Mohri, M. (2008). An efficient reduction of ranking to classification. In *Proceedings of the 21st annual conference on learning theory (COLT 2008)*. Helsinki: Omnipress.

Alon, N. (2006). Ranking tournaments. *SIAM Journal Discrete Mathematics*, *20*, 137–142.

Ariely, D., Loewenstein, G., & Prelec, D. (2008). Coherent arbitrariness: stable demand curves without stable preferences. *The Quarterly Journal of Economics*, *118*, 73–105.

Arrow, K. J. (1950). A difficulty in the concept of social welfare. *Journal of Political Economy*, *58*, 328–346.

Balcan, M.-F., Bansal, N., Beygelzimer, A., Coppersmith, D., Langford, J., & Sorkin, G. B. (2007). Robust reductions from ranking to classification. In *COLT* (pp. 604–619). Berlin: Springer.

Balcan, M.-F., Bansal, N., Beygelzimer, A., Coppersmith, D., Langford, J., & Sorkin, G. B. (2008). Robust reductions from ranking to classification. *Machine Learning Journal*, *72*, 139–153.

Clémençon, S., & Vayatis, N. (2007). Ranking the best instances. *Journal of Machine Learning Research*, *8*, 2671–2699.

Cohen, W. W., Schapire, R. E., & Singer, Y. (1999). Learning to order things. *The Journal of Artificial Intelligence Research*, *10*, 243–270.

Coppersmith, D., Fleischer, L., & Rudra, A. (2006). Ordering by weighted number of wins gives a good ranking for weighted tournaments. In *Proceedings of the 17th annual ACM-SIAM symposium on discrete algorithms (SODA)*.

Cortes, C., Mohri, M., & Rastogi, A. (2007a). An alternative ranking problem for search engines. In *Proceedings of the 6th workshop on experimental algorithms* (WEA 2007) (pp. 1–21). Heidelberg: Springer-Verlag.

Cortes, C., Mohri, M., & Rastogi, A. (2007b). Magnitude-preserving ranking algorithms. In *Proceedings of the twenty-fourth international conference on machine learning (ICML 2007)*. Oregon State University, Corvallis, OR.

Cossock, D., & Zhang, T. (2006). Subset ranking using regression. In *COLT* (pp. 605–619).

Crammer, K., & Singer, Y. (2001). Pranking with ranking. In *Advances in neural information processing systems: Vol. 14. Neural information processing systems: natural and synthetic, NIPS 2001* (pp. 641–647). Vancouver, British Columbia, Canada, December 3–8, 2001. Cambridge: MIT Press.

Freund, Y., Iyer, R. D., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, *4*, 933–969.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, *143*, 29–36.

Hedge, R., Jain, K., Williamson, D. P., & van Zuylen, A. (2007). Deterministic pivoting algorithms for constrained ranking and clustering problems. In *Proceedings of the ACM-SIAM symposium on discrete algorithms (SODA)*.

Hoare, C. (1961). Quicksort: Algorithm 64. *Communications of the ACM*, *4*, 321–322.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 133–142). New York: ACM Press.

Kenyon-Mathieu, C., & Schudy, W. (2007). How to rank with few errors. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on theory of computing* (pp. 95–103). New York: ACM Press.

Lehmann, E. L. (1975). *Nonparametrics: statistical methods based on ranks*. San Francisco: Holden-Day.

Montague, M. H., & Aslam, J. A. (2002). Condorcet fusion for improved retrieval. In *Proceedings of the 2002 ACM CIKM international conference on information and knowledge management* (pp. 538–548). McLean, VA, USA, November 4–9, 2002. New York: ACM.

Rudin, C., Cortes, C., Mohri, M., & Schapire, R. E. (2005). Margin-based ranking meets boosting in the middle. In *Learning theory, 18th annual conference on learning theory, COLT 2005 Proceedings* (pp. 63–78). Bertinoro, Italy, June 27–30, 2005. Berlin: Springer.

Williamson, D. P., & van Zuylen, A. (2007). Deterministic algorithms for rank aggregation and other ranking and clustering problems. In *Proceedings of the 5th workshop on approximation and online algorithms (WAOA)*.