Robust Reductions from Ranking to Classification

Maria-Florina Balcan[1], Nikhil Bansal[2], Alina Beygelzimer[2],
Don Coppersmith[3], John Langford[4], and Gregory B. Sorkin[2]

[1] Carnegie Melon University, Pittsburgh, PA
`ninamf@cs.cmu.edu`
[2] IBM Thomas J. Watson Research Center, Yorktown Heights + Hawthorne, NY
`{bansal,beygel,sorkin}@us.ibm.com`
[3] IDA Center for Communications Research, Princeton, NJ
`dcopper@idaccr.org`
[4] Yahoo Research, New York, NY
`jl@yahoo-inc.com`

Abstract. We reduce ranking, as measured by the Area Under the Receiver Operating Characteristic Curve (AUC), to binary classification. The core theorem shows that a binary classification regret of r on the induced binary problem implies an AUC regret of at most $2r$. This is a large improvement over approaches such as ordering according to regressed scores, which have a regret transform of $r \mapsto nr$ where n is the number of elements.

1 Introduction

We consider the problem of ranking a set of instances. In the most basic version, we are given a set of unlabeled instances belonging to two classes, 0 and 1, and the goal is to rank all instances from class 0 before any instance from class 1. A common measure of success for a ranking algorithm is the area under the ROC curve (AUC). The associated loss, $1 - \text{AUC}$, measures how many pairs of neighboring instances would have to be swapped to repair the ranking, normalized by the number of 0s times the number of 1s. The loss is zero precisely when all 0s precede all 1s; one when all 1s precede all 0s. It is greater for mistakes at the beginning and the end of an ordering, which satisfies the intuition that an unwanted item placed at the top of a recommendation list should have a higher associated loss than when placed in the middle.

The classification problem is simply predicting whether a label is 0 or 1 with success measured according to the error rate, i.e., the probability of a misprediction.

These two problems appear quite different. For the classification loss function, a misclassified instance incurs the same loss independently of how other instances are classified. The AUC loss, on the other hand, depends on the whole (ranked) sequence of instances. It is natural to ask whether we need fundamentally different algorithms to optimize these two loss functions. This paper shows that, in some precise sense, the answer is no. We prove that the problem of optimizing the AUC can be reduced to classification in such a way that good performance on the classification problem implies good performance on the AUC problem. We call a pair of instances *mixed* if they have different labels. The classification problem is to predict, given a random mixed pair of instances in the test set, whether the first

instance should be ordered before the second. In other words, we show that there is a robust mechanism for translating any binary classifier learning algorithm into a ranking algorithm.

Several observations should help understand the setting and the result better.

Relation to Regression and Classification: A common way to generate a ranking is to order examples according to a regressed score or estimated conditional class probability. The problem with this approach is that it is not robust (see, however, a discussion in Section 6). The fundamental difficulty is exhibited by highly unbalanced test sets. If we have one 1 and many 0s, a pointwise (i.e., regression or classification) loss on the 1 with a perfect prediction for the 0s can greatly harm the AUC while only slightly affecting the pointwise loss with respect to the induced distribution. This observation implies that such schemes transform pointwise loss l to AUC loss nl, where n is the number of elements in the test set.

A similar observation holds for regrets in place of losses: pointwise regret r translates into AUC regret nr. *Regret* is the difference between the incurred loss and the lowest achievable loss on the problem. The motivation for regret analysis is to separate avoidable loss from noise intrinsic to the problem, to give bounds that apply nontrivially even for problems with large intrinsic, stochastic noise.

Our core theorem (Theorem 1) shows that a pairwise classifier with regret r implies an AUC regret of at most $2r$, for arbitrary distributions over instances. Thus, for example, if the binary error rate is 20% due to inherent noise and 5% due to errors made by the classifier, then AUC regret is at most 10%, i.e., only the 5% would be at most doubled.

Section 5 shows that this is the best possible. The theorem is a large improvement over the approaches discussed above, which have a dependence on n. For comparison, the relationship of ranking to classification is functionally tighter than has been proven for regression to binary classification ($r \mapsto \sqrt{r}$) [LZ05].

Relation to the Feedback Arc Set Problem: Let U be the set of unlabeled examples we want to rank. There is a hidden bipartition of U into a set of 0s (called "winners") and a set of 1s (called "losers"), drawn from the underlying conditional distribution of label sequences given U.

Consider running a tournament on U. Every element (or "player" or "instance") of U plays all other elements, and the outcome of each play is determined by a classifier c trained to predict which of the two given elements should be ordered first. What is the best way to rank the players in U so that all winners are ordered before all losers?

The tournament induced by c on U does not have to be consistent with any linear ordering, while a ranking algorithm must predict an ordering. A natural objective is to find an ordering which agrees with the tournament on as many player pairs as possible, i.e., minimizes the number of inconsistent pairs where a higher-ranked player (one ordered closer to the beginning of the list) lost to a lower-ranked player. This is the NP-hard "minimum feedback arc set problem in tournaments". (Although the hardness was conjectured for a long time, it was proved only recently; see [A06].)

A *mistake* is defined as a winner–loser pair where the loser beats (i.e., is preferred to) the winner. A basic guarantee holds for a solution to the feedback arc set problem: If the classifier c makes at most k mistakes on U, then the algorithm minimizing the number of inconsistent pairs produces an ordering, or equivalently a transitive tournament, with at most $2k$ mistakes on U (see Section 4 for a proof); Section 5 exhibits a tournament matching this bound.

Instead of solving feedback arc set, another natural way to break cycles is to rank instances according to their number of wins in the tournament produced by c. The way ties are broken is inessential; for definiteness, let us say they are broken against us. Coppersmith, Fleischer, and Rudra [CFR06] proved that this algorithm provides a 5-approximation for the feedback arc set problem. An approximation, however, does not generally imply any finite regret transform for the AUC problem. For example, c may make no mistakes (i.e., make correct predictions on all winner–loser pairs) while inducing a non-transitive tournament on the winners or the losers, so an approximation that does not know the labeling can incur a non-zero number of mistakes.

We prove, however, that the algorithm that simply orders the elements by their number of wins, transforms classification regret k into AUC regret at most $2k$. That is, ordering by the number of wins has the *same* regret and loss transform as an optimal solution to the (NP-hard) feedback arc set problem. (Again, Section 5 shows that solving feedback arc set does no better.)

Relation to Generalization Bounds: A number of papers analyze generalization properties of ranking algorithms (see, e.g., [FIS+03,AHR05,AN05,RCM+05]). These results analyze ranking directly by estimating the rate of convergence of empirical estimates of the ranking loss to its expectation. The bounds typically involve some complexity parameter of the class of functions searched by the algorithms (which serves as a regularizer), and some additional quantities considered relevant for the analysis. The examples are assumed to be drawn independently from some fixed distribution.

The type of results in this paper is different. We bound the realized AUC performance in terms of the realized classification performance. Since the analysis is relative, it does not have to rely on any assumptions about the way the world produces data. In particular, the bounds apply when there are arbitrary high-order dependencies between examples. This seems important in a number of real-world applications where ranking is of interest.

By itself, our analysis does not say anything about the number of samples needed to achieve a certain level of performance. Instead it says that achieved performance can be robustly transferred from classification to ranking; and thus any generalization result for the induced classification problem immediately implies a generalization result for the AUC problem. For example, one can apply the bounds of Clémençon, Lugosi, and Vayatis [CLV05] on the performance of empirical risk minimization for the classification problem of ranking pairs of instances (under the assumption that training instances are independent and identically distributed).

The results of Clémençon, Lugosi, and Vayatis [CLV05] extend to the problem of ranking n independent instances (instead of just two), but the resulting bounds have

a dependence on n. The reduction presented here allows one to go from pairwise to total order robustly, i.e., achieve bounds that are *independent* of the number of instances.

2   Preliminaries

Classification: A *binary classification problem* is defined by a distribution P over $X \times \{0, 1\}$, where X is some observable feature space and $\{0, 1\}$ is the binary label space. The goal is to find a classifier $c : X \to \{0, 1\}$ minimizing the *classification loss* on P given by

$$e(c, P) = \mathbf{Pr}_{(x,y) \sim P}[c(x) \neq y].$$

The *classification regret* of c on P is defined as

$$r(c, P) = e(c, P) - \min_{c^*} e(c^*, P).$$

Ranking: Where $X^{(2)}$ denotes the set of ordered pairs of distinct elements of X, let $\pi : X^{(2)} \to \{0, 1\}$ be a *preference function* (called a *ranking rule* in [CLV05]): $\pi(x, x') = 1$ if π "prefers" x to x', and 0 otherwise (so $\pi(x, x') = 1 - \pi(x', x)$). If π is consistent with some linear ordering of a set of examples, we call π itself an *ordering* on this set. The *AUC loss* of an ordering π on a set $S = (x_1, y_1), \ldots, (x_n, y_n)$ is defined as

$$l(\pi, S) = \frac{\sum_{i \neq j} \mathbf{1}(y_i > y_j) \pi(x_i, x_j)}{\sum_{i < j} \mathbf{1}(y_i \neq y_j)}.$$

(Indices i and j in the summations range from 1 to n, and $\mathbf{1}(\cdot)$ is the indicator function which is 1 if its argument is true, and 0 otherwise.)

A pair of examples $(x_1, y_1), (x_2, y_2)$ is called *mixed* if $y_1 \neq y_2$.

An *AUC problem* is defined by a distribution D over $(X \times \{0, 1\})^*$. The goal is to find an ordering $\pi : X^{(2)} \to \{0, 1\}$ minimizing the expected AUC loss on D, given by

$$l(\pi, D) = \mathbf{E}_{S \sim D} l(\pi, S).$$

Note that D may encode arbitrary dependencies between examples. As an example, consider the internet search problem, where there is some underlying distribution of queries, each yielding a set of search results. This process generates a distribution over subsets; whether or not the subsets have the same size is inessential for the analysis.

The *AUC regret* of π on D is given by $r_{\mathrm{AUC}}(\pi, D) = l(\pi, D) - \min_{\pi^*} l(\pi^*, D)$. We are interested in ranking algorithms that use a preference function π as an oracle and are *robust* in the sense that a small classification regret of the oracle cannot induce a large AUC regret.

Tournaments: A *tournament* is a complete graph with no self-loops, in which each edge is directed one way or the other, so that for every pair of vertices $i \neq j$, either $i \to j$ is an edge or $j \to i$ is an edge, but not both. The edge $i \to j$ says that i *beats* j ("i is preferred to j"); edges point from winners to losers. We adopt

---

Algorithm 1   Auc-Train (labeled set S, binary learning algorithm A)

---

1. Let $S' = \{\langle (x_1, x_2), \mathbf{1}(y_1 < y_2) \rangle : (x_1, y_1), (x_2, y_2) \in S$ and $y_1 \neq y_2\}$
2. return $c = A(S')$.

Algorithm 2   Degree (unlabeled set U, pairwise classifier c)

---

1. For $x \in U$, let $\deg(x) = |\{x' : c(x, x') = 1, x' \in U\}|$.
2. Sort U in descending order of $\deg(x)$, breaking ties arbitrarily.

---

the convention that 0s should be ordered ahead of 1s, so 0s should ideally beat 1s. We write $\deg(i)$ for the *outdegree* of vertex i, so $\deg(i) = \sum_j \mathbf{1}(i \to j)$, where the indicator function $\mathbf{1}(i \to j)$ is 1 if $i \to j$ is an edge and 0 otherwise. Thus we generally expect 0s to have large outdegree and 1s small outdegree; however, we allow and analyze arbitrary tournaments.

3   Ordering by the Number of Wins

In this section, we describe the reduction and prove the main result.

The reduction consists of two components. The training part, Auc-Train (Algorithm 1), takes a set S of labeled examples of type $X \times \{0, 1\}$ and transforms all mixed pairs in S into binary examples for the oracle learning algorithm. The binary classification problem induced by the reduction is to predict, given a random mixed pair of examples in S, whether the first example should be ordered before the second. For any process D generating datasets S, we can define the induced distribution over $(X \times X) \times \{0, 1\}$ by first drawing S from D, and then drawing a random mixed pair from S. We denote this induced distribution by Auc-Train(D) (admittedly overloading the notation).

The test part, Degree (Algorithm 2), uses the pairwise classifier c learned in Algorithm 1 to run a tournament on a test set U, and then ranks the elements of U in decreasing order of their number of wins in the tournament, breaking ties arbitrarily. (Again, we expect 0s to beat 1s, and thus have larger outdegree.)

It is best to think of the classifier c as an adversary trying to induce a large AUC regret without paying much in classification regret: The adversary c specifies a tournament on U. There is some realized bipartition of U into a set of 0s and a set of 1s, drawn from the underlying conditional distribution of label sequences given U. The bipartition is known to the adversary but unknown to the ranking algorithm. The adversary starts with a tournament of its choice where every 0 beats every 1. She can invert the outcome of any game between a 0 and a 1, and she is charged for each such "mistake". Again, she can choose any subtournaments on the 0s and on the 1s for free. Unless we are extremely lucky, the resulting tournament is not consistent with any linear ordering of U.

Without seeing the labels, we wish to approximate c's tournament with a transitive tournament (or equivalently, a linear order) minimizing the number of mistakes

(i.e., pairs where a 1 precedes a 0 in the order). If c were itself consistent with a linear order, we could simply output that, at a cost in mistakes identical to the adversary's. In general it is not, and we would expect the cost of the more-constrained linear order to be higher. Our goal is to show that the reduction is robust in the sense that c cannot cause DEGREE to make many mistakes without making many mistakes itself: DEGREE makes at most twice as many mistakes as c.

The remainder of this section proves the following theorem.

Theorem 1. *For all distributions D and all pairwise classifiers c,*

$$r_{\mathrm{AUC}}(\mathrm{DEGREE}(\cdot, c), D) \leq 2r(c, \mathrm{AUC\text{-}TRAIN}(D)). \tag{1}$$

Note the quantification in the above theorem: it applies to *all* settings where Algorithms 1 and 2 are used; in particular, D may encode arbitrary dependences between examples.

Proof. Given an unlabeled test set $U \in X^n$, the joint distribution D induces a conditional distribution $D(Y_1, \ldots, Y_n \mid U)$ over the set of label sequences $\{0, 1\}^n$. In the remainder of the paper, let Q denote this conditional distribution. We identify U with $\{1, \ldots, n\}$. We prove the theorem by fixing U, taking the expectation over the draw of U at the end.

Our goal is to rewrite both sides of (1) as sums of pairwise regrets. A *pairwise loss* is defined by

$$l_Q(i, j) = \mathbf{E}_{y^n \sim Q} \frac{\mathbf{1}(y_i > y_j)}{\sum_{u < v} \mathbf{1}(y_u \neq y_v)}.$$

It is the loss of ordering i before j. If $l_Q(i, j) < l_Q(j, i)$, the *regret* $r_Q(i, j)$ of ordering i before j is 0; otherwise, $r_Q(i, j) = l_Q(i, j) - l_Q(j, i)$.

We can assume without loss of generality that the ordering minimizing the AUC loss (thus having zero AUC regret) is $\langle 1, 2, \ldots, n \rangle$. All regret-zero pairwise predictions must be consistent with the ordering; i.e., for all $i < j$, we have $r_Q(i, j) = 0$. (See Lemma 2 in Section A for a proof.)

Lemma 1 in Appendix A establishes a basic property of pairwise regrets: For any pair $i < j$, the regret $r_Q(j, i)$ can be decomposed as

$$r_Q(j, i) = \sum_{k=i}^{j-1} r_Q(k + 1, k).$$

This allows us to decompose the AUC regret of π on Q as a sum of pairwise regrets:

$$r_{\mathrm{AUC}}(\pi, Q) = l(\pi, Q) - \min_{\pi^*} l(\pi^*, Q) = \mathbf{E}_{y^n \sim Q}[l(\pi, U)] - \min_{\pi^*} \mathbf{E}_{y^n \sim Q}[l(\pi^*, U)] =$$

$$= \mathbf{E}_{y^n \sim Q} \frac{\sum_{i,j} \mathbf{1}(y_i > y_j)\pi(i,j)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)} - \min_{\pi^*} \mathbf{E}_{y^n \sim Q} \frac{\sum_{i,j} \mathbf{1}(y_i > y_j)\pi^*(i,j)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)}$$

$$= \max_{\pi^*} \mathbf{E}_{y^n \sim Q} \frac{\sum_{i,j}[\mathbf{1}(y_i > y_j)\pi(i,j) - \mathbf{1}(y_i > y_j)\pi^*(i,j)]}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)}$$

$$= \sum_{i<j:\pi(j,i)=1} r_Q(j,i) = \sum_{k=1}^{n-1} |\{i \leq k < j : \pi(j,i) = 1\}| \cdot r_Q(k+1,k).$$

The last equality follows from repeated application of Lemma 1.

The classification regret can also be written in terms of pairwise regrets:

$$r(c, \text{Auc-Train}(Q)) = e(c, \text{Auc-Train}(Q)) - \min_{c^*} e(c^*, \text{Auc-Train}(Q))$$

$$= \max_{c^*} \mathbf{E}_{y^n \sim Q} \left[\frac{\sum_{i,j}[\mathbf{1}(y_i > y_j)c(i,j) - \mathbf{1}(y_i > y_j)c^*(i,j)]}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)} \right]$$

$$= \sum_{i<j:c(j,i)=1} r_Q(j,i) = \sum_{k=1}^{n-1} |\{i \leq k < j : c(j,i) = 1\}| \cdot r_Q(k+1,k).$$

Let g_k and f_k denote the coefficients with which the term $r_Q(k+1,k)$ appears in the above decompositions of $r_{\text{AUC}}(\pi, Q)$ and $r(c, \text{Auc-Train}(Q))$ respectively. To complete the proof it suffices to show that $g_k \leq 2f_k$ for each k.

Fix k and consider a bipartition of U into a set $\{1, \ldots, k\}$ of "winners" and a set $\{k+1, \ldots, n\}$ of "losers". In this terminology, g_k is the number of winner–loser pairs where the loser has at least as many wins as the winner, and f_k is the number of winner–loser pairs where the loser beats the winner (in the tournament induced by c on U). Theorem 2 below shows that $g_k \leq 2f_k$, completing this proof. ∎

Let T be a tournament and let the vertices of T be arbitrarily partitioned into a set W of "winners" and a set L of "losers". Call the triple (T, W, L) a winner–loser partitioned tournament, and denote it by $\mathbf{T}$. We will show that for any $\mathbf{T}$, the number of winner–loser pairs where the loser's degree is larger than or equal to the winner's, is at most twice the number of winner–loser pairs where the loser beats the winner. Formally, define two measures:

$$g(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\deg(\ell) \geq \deg(w)),$$

$$f(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\ell \to w).$$

Theorem 2. *For any winner–loser partitioned tournament* $\mathbf{T}$, $g(\mathbf{T}) \leq 2f(\mathbf{T})$.

Since the number of edges from L to W is equal to the total number of edges out of L minus the number of edges from L to L, we can rewrite

$$f(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\ell \to w) = \sum_{\ell \in L} \deg(\ell) - \binom{|L|}{2}.$$

Both $f(\mathbf{T})$ and $g(\mathbf{T})$ depend only on the degrees of the vertices of T, so rather than working with a (labeled) tournament, a relatively complex object, we can work with a (labeled) degree sequence.

Landau's theorem [Lan53] says that there exists a tournament with outdegree sequence $d_1 \leq d_2 \leq \cdots \leq d_n$ if and only if, for all $1 \leq i \leq n$, $\sum_{j=1}^{i} d_j \geq \sum_{j=1}^{i}(j-1)$, with equality for $i = n$.

Recall that a sequence $\langle a_1, \ldots, a_n \rangle$ is *majorized* by $\langle b_1, \ldots, b_n \rangle$ if the two sums are equal and if, when each sequence is sorted in non-increasing order, the prefix sums of the b sequence are at least as large as (dominate) those of the a sequence. (For a comprehensive treatment of majorization, see [MO79].) Landau's condition is precisely that $\langle d_1, \ldots, d_n \rangle$ is majorized by $\langle 0, \ldots, n-1 \rangle$. (With the sequences sorted in increasing order, Landau's condition is that prefix sums of the degree sequence dominate those of the progression, which is the same as saying that the suffix sums of the degree sequence are dominated by the suffix sums of the progression.) This allows us to take advantage of well-known properties of majorization, notably that if A' is obtained by averaging together any elements of A, then A majorizes A'.

This allows us to restate Theorem 2 in terms of a sequence and majorization, rather than a tournament, but first we relax the constraints. First, where the original statement requires elements of the degree sequence to be non-negative integers, we allow them to be non-negative reals. Second, the original statement requires that we attach a winner/loser label to each element of the degree sequence. Instead, we aggregate equal elements of the degree sequence, and for a degree d_i of (integral) multiplicity m_i, assign arbitrary non-negative (but not necessarily integral) portions to winners and losers: $w_i + \ell_i = m_i$.

Let $\mathbf{D} = (D, W, L)$ be such a generalized "winner–loser labeled compressed sequence". Note that the majorization condition applies only to the values $\{d_i, m_i\}$, not the labeling. The definitions of f and g above are easily extended to this broader domain:

$$g(\mathbf{D}) = \sum_i \sum_{j \leq i} l_i w_j, \qquad f(\mathbf{D}) = \sum_i l_i d_i - \binom{\sum_i l_i}{2},$$

where we define $\binom{x}{2} = x(x-1)/2$ for all x (not just integers). If we prove $g \leq 2f$ over this larger domain, the inequality holds in particular for plain winner–loser labeled degree sequences (the case where all weights happen to be integral). That is, Theorem 3, below, implies Theorem 2.

Theorem 3. *For any winner–loser labeled compressed sequence* $\mathbf{D} = (D, W, L)$ *where D is majorized by $\langle 0, \ldots, n-1 \rangle$, $g(\mathbf{D}) \leq 2f(\mathbf{D})$.*

Proof. We begin with an outline of the proof. Define a compressed sequence $\mathbf{D}$ as being *canonical* if it consists of at most three degrees: a smallest degree d_1 having only losers ($w_1 = 0$), a middle degree d_2 potentially with both winners and losers ($w_2, \ell_2 \geq 0$), and a largest degree d_3 having only winners ($\ell_3 = 0$). We first establish that any canonical sequence has $g(\mathbf{D}) - 2f(\mathbf{D}) \leq 0$. We then show how to transform *any* degree sequence to a canonical one with a larger (or equal) value of $g - 2f$, which completes the argument.

We first show that a canonical sequence $\mathbf{D}$ has $g - 2f \leq 0$. For the canonical configuration, $g = w_2 \ell_2$ and $f = \ell_1 d_1 + \ell_2 d_2 - \binom{\ell_1 + \ell_2}{2}$, and hence our goal is to show that

$$\ell_1 d_1 + \ell_2 d_2 \geq (\ell_1 + \ell_2)(\ell_1 + \ell_2 - 1)/2 + w_2 \ell_2 / 2 \tag{2}$$

By Landau's condition applied to ℓ_1 and $\ell_1 + w_2 + \ell_2$, we have the following two relations:

$$\ell_1 d_1 \geq \binom{\ell_1}{2} \tag{3}$$

and

$$\ell_1 d_1 + (\ell_2 + w_2) d_2 \geq \binom{\ell_1 + w_2 + \ell_2}{2}. \tag{4}$$

Multiplying (3) by $w_2/(\ell_2 + w_2)$ and (4) by $\ell_2/(\ell_2 + w_2)$ and adding them, we obtain that

$$\ell_1 d_1 + \ell_2 d_2 \geq \frac{1}{\ell_2 + w_2} \left(w_2 \binom{\ell_1}{2} + \ell_2 \binom{\ell_1 + \ell_2 + w_2}{2} \right). \tag{5}$$

A simple calculation shows that the right side of inequality (5) is exactly equal to the right hand side of (2). This proves that $g \leq 2f$ for a canonical sequence.

If a sequence is not canonical then there are two consecutive degrees d_i and d_j ($j = i + 1$) such that one of the cases 1a, 1b, or 2 (described below) holds. In each case we apply a transformation producing from the degree sequence $\mathbf{D}$ a new degree sequence $\mathbf{D}'$, where:

- the total weight of winners in $\mathbf{D}'$ is equal to that of $\mathbf{D}$; similarly for losers, and thus for the total weight; furthermore, the total weight on each degree remains integral;

- $\mathbf{D}'$ maintains the majorization needed for Landau's theorem;

- the value of $g - 2f$ is at least as large for $\mathbf{D}'$ as for $\mathbf{D}$; and

- either the number of nonzero values w_i and ℓ_i or the number of distinct degrees d_i is strictly smaller for $\mathbf{D}'$ than for $\mathbf{D}$, and the other is no larger for $\mathbf{D}'$ than for $\mathbf{D}$.

We first sketch the cases and then detail the transformations.

Case 1a d_i has only winners ($l_i = 0$).
    Apply Transformation 1a, combining the two degrees into one.

Case 1b d_j has only losers ($w_j = 0$).
    Apply Transformation 1b, combining the two degrees into one.

Case 2 All of w_i, l_i, w_j and l_j are nonzero.
    Apply Transformation 2, leaving the degrees the same but transforming the weights so that one of them is equal to 0 and one of the preceding cases applies, or the weights obey an equality allowing application of Transformation 3, which combines the two degrees into one.

Either there is some pair i, j to which one of the cases applies, or the sequence is canonical. We argue this by showing that if there is no pair i, j for which Cases 1a or 1b apply, then either the sequence is canonical, or there is a pair to which Case 2 applies. First, note that for any $i \neq n$, $l_i > 0$ (else Case 1a applies to $i, i+1$) and for any $i \neq 1$, $w_i > 0$ (else Case 1b applies to $i-1, i$). In particular, for any $1 < i < n$, both $l_i, w_i > 0$. If $n \geq 4$ this implies immediately that Case 2 applies to the pair $2, 3$. If $n = 1$, $\mathbf{D}$ is automatically canonical. If $n = 2$ and $l_2 = 0$ or $w_1 = 0$ then $\mathbf{D}$ is canonical, while if both $l_2, w_1 > 0$ we may apply Case 2 (since, as we first argued, $l_1, w_2 > 0$). Finally, if $n = 3$, we know $l_1, l_2, w_2, w_3 > 0$. If $w_1 = l_3 = 0$ then $\mathbf{D}$ is canonical, and otherwise Case 2 applies.

Transformation 1a: In Case 1a, where d_i has only winners, change $\mathbf{D}$ to a new sequence $\mathbf{D}'$ by replacing the pair $(d_i, w_i, 0)$, (d_j, w_j, l_j) by their "average": the single degree (d', w', l'), where

$$w' = w_i + w_j, \quad l' = l_j, \quad d' = \frac{w_i d_i + (w_j + l_j) d_j}{w_i + w_j + l_j}.$$

The stated conditions on a transformation are easily checked. The total weight of winners is clearly preserved, as is the total weight of losers and the total degree (out-edges). Summing weights preserves integrality. The number of distinct degrees is reduced by one, and the number of nonzero weights may be decreased by one or may remain unchanged. The Landau majorization condition holds because $\mathbf{D}'$, as an averaging of $\mathbf{D}$, is majorized by it, and majorization is transitive. The only non-trivial condition is the non-decrease in $g - 2f$. The number of loser–winner pairs where the loser outranks the winner remains the same, so $g(\mathbf{D}) = g(\mathbf{D}')$. Also, f depends only on the total weight of losers (which is unchanged) and on the average degree of losers. This average degree would be unchanged if w_i were 0; since $w_i \geq 0$, the average degree may decrease. Thus $f(\mathbf{D}) \geq f(\mathbf{D}')$, and $(g - 2f)(\mathbf{D}) \leq (g - 2f)(\mathbf{D}')$, as desired.

Transformation 1b: Symmetrically to Transformation 1a, obtain $\mathbf{D}'$ by replacing the pair of labeled weighted degrees (d_i, w_i, l_i) and $(d_j, 0, l_j)$ with a single one (d', w', l'), where $w' = w_i$, $l' = l_i + l_j$, and $d' = [(l_i + w_i) d_i + l_j d_j]/(l_i + w_i + l_j)$.

Transformation 2: Where w_i, l_i, w_j and l_j are all nonzero, we begin with one case, which leads to one other. In the usual case, we transform $\mathbf{D}$ to $\mathbf{D}'$ by replacing the pair (d_i, w_i, l_i), (d_j, w_j, l_j) with $(d_i, w_i + x, l_i - x)$, $(d_j, w_j - x, l_j + x)$, for some value of x (positive or negative) to be determined. This affects only the labeling, not the weighted degree sequence itself, and is therefore legitimate as long as the four quantities $w_i + x$, $l_i - x$, $w_j - x$ and $l_j + x$ are all non-negative.

Defining $\Delta = (g - 2f)(\mathbf{D}') - (g - 2f)(\mathbf{D})$, we wish to choose x to make $\Delta > 0$.

$$\Delta = \left\{ \left[(l_j + x)(w_i + x + w_j - x) + (l_i - x)(w_i + x) \right] - \left[l_j(w_i + w_j) + l_i w_i \right] \right\}$$
$$- 2\left\{ \left[(l_i - x) d_i + (l_j + x) d_j \right] - \left[l_i d_i + l_j d_j \right] \right\}$$
$$= x(w_j + l_i - 2(d_j - d_i) - x) = x(a - x),$$

where $a = w_j + l_i - 2(d_j - d_i)$. This is a simple quadratic expression with negative coefficient on x^2, so its value increases monotonically as x is varied from 0 to $a/2$,

where the maximum is obtained. (Note that a may be negative.) If $a = 0$ then we do not use this transformation but Transformation 3, below. Otherwise, vary x from 0 to $a/2$ stopping when x reaches $a/2$ or when any of $w_i + x$, $l_i - x$, $w_j - x$ and $l_j + x$ becomes 0. Call this value $x^\star$, and use it to define the transformation.

If any of $w_i + x$, $l_i - x$, $w_j - x$ and $l_j + x$ is 0 then the number of nonzero weights is decreased (while the number of distinct degees is unchanged). Otherwise, $x^\star = a/2$. In that case, the new $\mathbf{D}'$ has $a = 0$ (the optimal "weight shift" has already been performed). With $a = 0$ we apply Transformation 3, which reduces the number of nonzero weights.

Transformation 3: Similar to Cases 1a and 1b, transform $\mathbf{D}$ to $\mathbf{D}'$ by replacing the pair (d_i, w_i, l_i), (d_j, w_j, l_j) with a single degree (d', w', l') that is their weighted average,

$$w' = w_i + w_j, \quad l' = l_i + l_j, \quad d' = \frac{(w_i + l_i)d_i + (w_j + l_j)d_j}{w_i + l_i + w_j + l_j}.$$

This gives

$$\begin{aligned}
\Delta &= (g - 2f)(\mathbf{D}') - (g - 2f)(\mathbf{D}) \\
&= (l_i w_j) - 2(l_i d' + l_j d' - l_i d_i - l_j d_j) \\
&= l_i w_j + \frac{2(d_j - d_i)(w_i l_j - w_j l_i)}{w_i + l_i + w_j + l_j}.
\end{aligned}$$

We apply this transformation only in the case where Transformation 2 fails to give any improvement because its "a" expression is equal to 0, i.e., $d_j - d_i = (w_j + l_i)/2$. Making the corresponding substitution gives

$$\begin{aligned}
\Delta &= l_i w_j + \frac{(w_j + l_i)(w_i l_j - w_j l_i)}{w_i + l_i + w_j + l_j} \\
&= \frac{(l_i w_j)(l_j + w_i) + (l_j w_i)(l_i + w_j)}{w_i + l_i + w_j + l_j} > 0.
\end{aligned}$$

This reduces the number of distinct degrees by one, without increasing the number of nonzero weights.

Concluding the argument, we have shown that any non-canonical configuration $\mathbf{D}$ can be replaced by a configuration with a strictly smaller total of distinct degrees and nonzero weights, and at least as large a value of $g - 2f$. Since $\mathbf{D}$ had at most n distinct degrees and $2n$ nonzero weights originally, a canonical configuration $\mathbf{D}^\star$ is reached after at most $3n - 1$ transformations. (All that is important is that the number of transformations is finite: that a canonical configuration is eventually reached.) Then, $(g - 2f)(\mathbf{D}) \leq (g - 2f)(\mathbf{D}^\star) \leq 0$. ∎

A further generalization of Theorem 3 may be found in [BCS06].

4   An Upper Bound for Minimum Feedback Arc Set

This section shows an analog of Theorem 2 for an optimal solution to the feedback arc set problem. (The decomposition argument in Theorem 1 is algorithm-independent and applies here as well.) For a tournament T and an ordering π, a

back edge is an edge $i \to j$ in T such that j is ordered before i in π. Let $\mathrm{back}(T, \pi)$ denote the number of back edges induced by π in T.

For a winner–loser partitioned tournament $\mathbf{T} = (T, W, L)$ and any minimum feedback arc set ordering π of T, let $g'(\mathbf{T}, \pi)$ be the number of winner–loser pairs where the loser comes before the winner in π, and as before let

$$f(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\ell \to w)$$

be the number of winner–loser pairs where the loser beats the winner.

Theorem 4. *For any winner–loser partitioned tournament $\mathbf{T} = (T, W, L)$ and any minimum feedback arc set ordering π of T, $g'(\mathbf{T}, \pi) \leq 2f(\mathbf{T})$.*

Proof. Let k_w be the smallest possible number of back edges in the subtournament induced by W. Define k_l similarly for the subtournament induced by L. Let k_w^π and k_l^π be the number of back edges in π that go from W to W and from L to L, respectively. Denote the number of remaining (i.e., winner–loser or loser–winner) back edges in π by k_o^π.

Consider another ordering σ where all winners are ordered before all losers, and both the winners and the losers are ordered optimally among themselves, i.e., with k_w and k_l back edges respectively. The number of back edges in σ is $\mathrm{back}(\mathbf{T}, \sigma) = k_w + k_l + f(\mathbf{T})$. But we also have $\mathrm{back}(\mathbf{T}, \sigma) \geq \mathrm{back}(\mathbf{T}, \pi)$ since π minimizes the number of back edges, and thus $k_w + k_l + f(\mathbf{T}) \geq k_w^\pi + k_l^\pi + k_o^\pi$. Since $k_w \leq k_w^\pi$ and $k_l \leq k_l^\pi$ by definition of k_w and k_l, we have $f(\mathbf{T}) \geq k_o^\pi$.

Consider any winner–loser pair with the loser ordered before the winner. If $w \to l$ is the edge, it is a back edge in π and thus is counted by k_o^π. If $l \to w$ is the edge instead, it is counted by $f(\mathbf{T})$. Thus $g'(\mathbf{T}, \pi)$ is at most $k_o^\pi + f(\mathbf{T})$. Since $f(\mathbf{T}) \geq k_o^\pi$, this number is never more than $2f(\mathbf{T})$, which implies $g'(\mathbf{T}, \pi) \leq 2f(\mathbf{T})$. ∎

5 Lower Bounds

We first show that Theorem 2 is best possible: the DEGREE ranking really can make twice as many mistakes as the adversary. Recall that f denotes the number of winner–loser pairs where the loser beats the winner, and g the number of winner–loser pairs where the loser outranks the winner. The example below generates an infinite family of tournaments with $g = 2f$.

Example 1. With n odd, let every vertex have degree $(n-1)/2$; note that the degree sequence $\langle \frac{n-1}{2}, \ldots, \frac{n-1}{2} \rangle$ does indeed respect Landau's condition, so it is realizable as a tournament. Label $(n-1)/2$ of the vertices as winners and $(n+1)/2$ as losers. With ties broken against us, all winners are ordered after all losers. This gives $f = \frac{n+1}{2} \cdot \frac{n-1}{2} - \binom{(n+1)/2}{2} = (n+1)(n-1)/8$, while $g = \frac{n+1}{2} \cdot \frac{n-1}{2} = (n+1)(n-1)/4 = 2f$. (A similar example gives a lower bound of $2 - O(1/n)$ with ties broken optimally.) ∎

Theorem 4 is also essentially best possible. The next construction gives an infinite family of tournaments for which an optimal solution to the feedback arc set problem has $g \geq (2 - \epsilon)f$, for any $\epsilon > 0$.

Example 2. Set $\delta = \frac{\epsilon}{1-\epsilon}$, and let the set of vertices be partitioned into three components, V_1, V_2, and V_3, with $|V_1| = \delta n^2$, $|V_2| = 2n^2$, and $|V_3| = n$, for a sufficiently large n. The vertices in $V_1 \cup V_2$ are the winners, the vertices in V_3 are the losers.

The edges within each of the three components form acyclic tournaments. The cross-component edges are defined as follows: All edges between V_3 and V_1 point from V_3 to V_1, and all edges between V_1 and V_2 point from V_1 to V_2. To define the edges between V_2 and V_3, divide V_2 into $2n$ consecutive blocks $B_1, \ldots, B_{2n}$ of n vertices each, such that edges point from B_i to B_j where $1 \leq i < j \leq 2n$. If i is odd, all edges from B_i point to V_3; otherwise all edges from V_3 point to B_i.

We have $f = (1 + \delta)n^3$. What is the value of g induced by an ordering minimizing the number of back edges? Any such ordering must put V_1 before V_2; otherwise we would have $\Omega(n^4)$ back edges while an optimal ordering does not need to have more than $O(n^3)$ such edges. Now, the tournament induced by $V_2 \cup V_3$ has n^3 edge-disjoint cycles of length 3 since there are n^2 such cycles for every pair of blocks in V_2 (and there are n disjoint pairs). There has to be at least one back edge for every such cycle, so any ordering must have at least n^3 back edges. The ordering that puts V_3 before V_2 is thus optimal since it has exactly n^3 back edges. Thus the ordering $\{V_3, V_1, V_2\}$ minimizes the number of back edges. This ordering has $(2 + \delta)n^3$ pairs where the loser is ordered before the winner, implying the bound $g \geq (2 - \frac{\delta}{1+\delta})f = (2 - \epsilon)f$. ∎

6   Practicality and Relation to Other Work

The reduction analysis is representation independent, which implies that it works for any representation. Naturally, some representations are more computationally efficient than others. If, for example, $c(x_i, x_j) = \mathbf{1}(s(x_i) > s(x_j))$ for some learned score function $s : X \to [0, 1]$, the complexity of test-time evaluation is linear rather than quadratic in the number of elements. Note that s is not trained as a simple regressor, because what we want to optimize is the pairwise ordering of elements (see [CLV05]).[5]

Cohen, Schapire, and Singer [CSS99], similarly, use a two-stage approach to ranking: They first learn a preference function that takes a pair of instances and returns a score predicting how certain it is that the first instance should be ranked before the second. The learned function is then evaluated on all pairs of instances in the test set and an ordering approximating the largest possible l_1 agreement with the

---

[5] Note that the minimum in the definition of classification regret is over all possible preference functions. The result also holds if c is constrained to be in some class of functions, as long as the class contains an optimal classifier minimizing the loss on the induced problem. For the case of binary labels, the optimal classifier can clearly be represented by a scoring function [CLV05].

predictions is created, using a variant of the degree-based algorithm. One of the results they show is that the agreement achieved by an optimal feedback arc set ordering is at most twice the agreement obtained by their algorithm. To translate this result into the language of losses, let MFA be the AUC loss of the minimum feedback arc set ordering and APPROX be the AUC loss of the approximation. Then the result says that $1 - \text{APPROX} \geq \frac{1}{2}(1 - \text{MFA})$ or $\text{APPROX} \leq \frac{1}{2} + \text{MFA}/2$. The result is difficult to compare with the results given here, as the settings are different. A very rough comparison requires specializations and yields a bound that is weaker than ours: As we have seen in Section 4, $\text{MFA} \leq 2\,\text{BIN}$, where BIN is the loss of the pairwise predictor, so the result of [CSS99] roughly says that $\text{APPROX} \leq \frac{1}{2} + \text{BIN}$, while we show that $\text{APPROX} \leq 2\,\text{BIN}$ (modulo the slight differences in the approximation algorithm and the binary problem).

Cortes and Mohri [CM04] analyzed the relationship between the AUC and the error rate on the same classification problem, treating the two as different loss functions. They derived expressions for the expected value and the standard deviation of the AUC over all classifications with a fixed number of errors, under the assumption that all such classifications are equiprobable (i.e., the classifier is as likely to err on any one example as on any other). There is no direct connection with the present work.

References

[AHR05]   S. Agarwal, S. Har-Peled, and D. Roth. A uniform convergence bound for the area under the ROC curve, *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, 2005.

[AN05]   S. Agarwal, P. Niyogi. Stability and Generalization of Bipartite Ranking Algorithms, *Proceedings of the Eighteenth Annual Conference on Computational Learning Theory* (COLT), 32–47, 2005.

[A06]   N. Alon. Ranking tournaments, *SIAM Journal on Discrete Mathematics* **20**: 137–142, 2006.

[BCS06]   N. Bansal, D. Coppersmith, and G.B. Sorkin. A Winner–Loser Labeled Tournament Has at Most Twice as Many Outdegree Misrankings as Pair Misrankings, *IBM Research Report RC24107*, Nov. 2006.

[CLV05]   S. Clémençon, G. Lugosi, N. Vayatis. Ranking and Scoring Using Empirical Risk Minimization, *Proceedings of the Eighteenth Annual Conference on Computational Learning Theory* (COLT), 1–15, 2005.

[CSS99]   W. Cohen, R. Schapire, and Y. Singer. Learning to order things, *Journal of Artificial Intelligence Research* **10**: 243–270, 1999.

[CFR06]   D. Coppersmith, L. Fleischer and A. Rudra. Ordering by Weighted Number of Wins Gives a Good Ranking for Weighted Tournaments. *Proceeding of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 776–782, 2006.

[CM04]   C. Cortes and M. Mohri. AUC Optimization vs. Error Rate Minimization, *Advances in Neural Information Processing Systems* (NIPS), 2004.

[FIS+03]   Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences, *Journal of Machine Learning Research* **4**: 933–969, 2003.

[Lan53]   H. G. Landau. On Dominance Relations and the Structure of Animal Societies: III. The Condition for a Score Structure, *Bull. Math. Biophys.*, **15**: 143–148, 1953.

[LZ05]   J. Langford and B. Zadrozny. Estimating Class Membership Probabilities Using Classifier Learners, *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, 2005.

[MO79]   A. Marshall and I. Olkin. *Inequalities: Theory of majorization and its applications*, Mathematics in Science and Engineering, 143, New York, 1979.

[RCM+05] C. Rudin, C. Cortes, M. Mohri, and R. Schapire. Margin-based ranking meets Boosting in the middle, *Proceedings of the Eighteenth Annual Conference on Computational Learning Theory* (COLT), 2005.

A Supporting Lemmas

The proof of Theorem 1 used two simple lemmas which we prove here. Recall that Q denotes the distribution of label sequences $\{0,1\}^n$ of an unlabeled set $\{1, \ldots, n\}$.

Lemma 1. *For any i, j, and k in $\{1, \ldots, n\}$, $r_Q(i,j) + r_Q(j,k) = r_Q(i,k)$.*

Proof. Let p be a shorthand for the restriction of Q to indices $\{i, j, k\}$ (so p is a distribution over $\{0,1\}^3$ obtained by summing over all label indices other than i, j, and k). A simple algebraic manipulation verifies the claim.

$$r_Q(i,j) + r_Q(j,k) =$$
$$p(100) + p(101) - p(010) - p(011) + p(010) + p(110) - p(001) - p(101) =$$
$$p(100) + p(110) - p(001) - p(011) = r_Q(i,k).$$

Notice that all label assignments above have exactly two mixed pairs, so the factor of $1/2$ is cancelled. ∎

Lemma 2. *If, with respect to Q, the ordering $\langle 1, 2, \ldots, n \rangle$ is the ordering minimizing the AUC loss (thus having zero AUC regret), then all regret-zero pairwise predictions must be consistent with the ordering; i.e., $r_Q(i,j) = 0$ for all $1 \le i < j \le n$.*

Proof. All regrets $r_Q(i, i+1)$ must be 0, since swapping i and $i+1$ does not affect other pairwise regrets and would thus decrease the overall regret, contradicting the assumption that $1\,2\,\ldots\,n$ is regret minimizing. We prove that all other pairwise regrets must be 0 recursively. Consider a triple, $i < j < k$ with $r_Q(i,j) = 0$ and $r_Q(j,k) = 0$. We show that $r_Q(i,k) = 0$. Let p be defined as in the proof of Lemma 1. We have $l_Q(i,j) \le l_Q(j,i)$ and $l_Q(j,k) \le l_Q(k,j)$, which can be expanded as

$$p(100) + p(101) \le p(010) + p(011), \quad p(010) + p(110) \le p(001) + p(101). \quad (6)$$

We now have

$$l_Q(i,k) = p(100) + p(110) \le p(100) + p(101) + p(010) + p(110)$$
$$\le p(010) + p(011) + p(001) + p(101) \le p(001) + p(011) = l_Q(k,i),$$

where the second inequality follows by adding the two inequalities in (6). Thus $l_Q(i,k) \le l_Q(k,i)$ implying $r_Q(i,k) = 0$. ∎