Perceptron-like Algorithms for Online Learning to Rank

Sougata Chaudhuri
Department of Statistics,
University of Michigan,
1085 S. University Ave., Ann Arbor, MI 48109, USA

Ambuj Tewari
Department of Statistics, and
Department of Electrical Engineering and Computer Science,
University of Michigan,
1085 S. University Ave., Ann Arbor, MI 48109, USA

Abstract

Perceptron is a classic online algorithm for learning a classification function. In this paper, we provide a novel extension of the perceptron algorithm to the learning to rank problem in information retrieval. We consider popular listwise performance measures such as Normalized Discounted Cumulative Gain (NDCG) and Average Precision (AP). A modern perspective on perceptron for classification is that it is simply an instance of online gradient descent (OGD), during mistake rounds, on the hinge loss function. Motivated by this interpretation, we propose a novel family of listwise, large margin ranking surrogates, which are adaptable to NDCG and AP measures. Exploiting a certain self-bounding property of the proposed family, we provide a guarantee on the cumulative NDCG/AP induced loss incurred by our perceptron-like algorithm. We show that, if there exists a perfect oracle ranker which can correctly rank, with some margin, each instance in an online sequence, the cumulative NDCG/AP induced loss of perceptron algorithm on that sequence is bounded by a constant, irrespective of the length of the sequence. This result is reminiscent of Novikoff’s convergence theorem for the classification perceptron and fills the analogous gap in learning to rank literature. Moreover, we prove a lower bound on the cumulative loss achievable by any deterministic algorithm, under the assumption of existence of perfect oracle ranker. The lower bound shows that our perceptron bound is not tight, and we propose another online algorithm which achieves the lower bound. Our experiments on simulated and large commercial datasets corroborate our theoretical results.

1. Introduction

Learning to rank is a supervised learning problem where the output space consists of rankings of a set of objects. In the learning to rank problem that frequently arises in information retrieval, the objective is to rank documents associated with a query, in the order of the relevance of the documents for the given query. The accuracy of a ranked list, given actual relevance scores of the documents, is measured by various ranking performance measures, such as Normalized Discounted Cumulative Gain (NDCG) [Järvelin and Kekäläinen, 2002], Average Precision (AP) [Baeza-Yates and Ribeiro-Neto, 1999] and others. Since optimization of ranking measures during training phase is computationally intractable, ranking methods are based on minimizing surrogate losses that are easy to optimize.

The historical importance of the perceptron algorithm in the classification literature is immense [Rosenblatt, 1958, Freund and Schapire, 1999]. Vapnik [1999] says, “In 1962 Novikoff proved the
first theorem about the perceptron (Novikoff, 1962). This theorem actually started learning theory” (emphasis added). Classically the perceptron algorithm was not linked to surrogate minimization but the modern perspective on perceptron is to interpret it as online gradient descent (OGD), during mistake rounds, on the hinge loss function [Shalev-Shwartz, 2011]. The hinge loss has special properties that allow one to establish bounds on the cumulative zero-one loss (viz., the total number of mistakes) in classification, without making any statistical assumptions on the data generating mechanism. Novikoff’s celebrated result about the perceptron says that, if there is a perfect linear classification function which can correctly classify, with some margin, every instance in an online sequence, then the total number of mistakes made by perceptron, on that sequence, is bounded.

Our work provides a novel extension of the perceptron algorithm to the learning to rank setting with focus on two listwise ranking measures, NDCG and AP. Listwise measures are so named because the quality of ranking function is judged on an entire list of document, associated with a query, with emphasis on errors near top of the ranked list. Akin to the cumulative 0-1 loss (or, equivalently, the total number of mistakes) bound of perceptron in classification, we develop bounds on cumulative losses induced by NDCG and AP. We prove that if there exists a perfect linear ranking function, which can correctly rank, with some margin, every instance in an online sequence, the cumulative loss of our perceptron is bounded, independent of the number of instances. The bound achieved, however, is sub-optimal, since it is dependent on number of documents per query. We prove a lower bound, on the cumulative loss achievable by any deterministic algorithm, which is independent of number of documents per query. We then propose a second perceptron type algorithm, which achieves the theoretical lower bound, but has worse empirical performance on real data.

We make the following contributions in this work.

- We develop a family of listwise large margin ranking surrogates. The family consists of Lipschitz functions and is parameterized by a set of weight vectors that makes them adaptable to losses induced by performance measures NDCG and AP. The family of surrogates is an extension of the hinge surrogate in classification that upper bounds the 0-1 loss. The family of surrogates has a special self-bounding property, where the norm of the gradient of a surrogate can be bounded by the surrogate loss itself.

- We exploit the self bounding property of the surrogates to develop an online perceptron-like algorithm for learning to rank (Algorithm 2). We provide bounds on the cumulative NDCG and AP induced losses (Theorem, 6). We prove that, if there is a perfect linear ranking function which can rank correctly, with some margin, every instance in an online sequence, our perceptron-like algorithm perfectly ranks all but a finite number of instances (Corrollary, 7). This implies that the cumulative loss induced by NDCG or AP is bounded by a constant, and can be seen as an extension of the classification perceptron mistake bound (Theorem, 1). Our perceptron bound, however, depends linearly on the number of documents per query. In practice, during evaluation, NDCG is often cut off at a point which is much smaller than number of documents per query. In that scenario, we prove that the cumulative NDCG loss of our perceptron is upper bounded by a constant which is dependent on the cut-off point. (Theorem, 8).

- We prove a lower bound, on the cumulative loss induced by NDCG or AP, that can be achieved by any deterministic online algorithm (Theorem, 9). The lower bound is independent of the
number of documents per query. We also prove that it is the minimax bound on cumulative loss, by proposing another perceptron type algorithm which achieves the bound (Algorithm 3 and Theorem 10). However, the surrogate on which the perceptron type algorithm operates is not listwise in nature and does not adapt to different performance measures. Thus, its empirical performance on real data is significantly worse than the first perceptron algorithm (Algorithm 2).

• We provide empirical results on simulated as well as large scale benchmark datasets and compare the performance of our perceptron algorithm with the online version of the widely used ListNet learning to rank algorithm [Cao et al., 2007].

The rest of the paper is organized as follows. Section 2 provides formal definitions and notations related to the problem setting. Section 3 provides a review of perceptron for classification, including algorithm and theoretical analysis. Section 4 introduces the family of listwise large margin ranking surrogates, and contrasts with a number of existing large margin ranking surrogates in literature. Section 5 introduces the perceptron algorithm for learning to rank, and compares and contrasts with the perceptron for classification. Section 6 establishes minimax bound on NDCG/AP induced cumulative loss. Section 7 compares our work with existing perceptron algorithms for ranking. Section 8 provide empirical results on simulated and large scale benchmark datasets.

2. Problem Definition

In learning to rank, we formally denote the input space as \( \mathcal{X} \subseteq \mathbb{R}^{m \times d} \). Each input consists of \( m \) rows of document-query features represented as \( d \) dimensional vectors. Each input corresponds to a single query and, therefore, the \( m \) rows have features extracted from the same query but \( m \) different documents. In practice \( m \) changes from one input instance to another but we treat \( m \) as a constant for ease of presentation. For \( X \in \mathcal{X} \), \( X = (x_1, \ldots, x_m)^\top \), where \( x_i \in \mathbb{R}^d \) is the feature extracted from a query and the \( i \)th document associated with that query. The supervision space is \( \mathcal{Y} \subseteq \{0, 1, \ldots, n\}^m \), representing relevance score vectors. If \( n = 1 \), the relevance vector is binary graded. For \( n > 1 \), relevance vector is multi-graded. Thus, for \( R \in \mathcal{Y} \), \( R = (R_1, \ldots, R_m)^\top \), where \( R_i \) denotes relevance of \( i \)th document to a given query. Hence, \( R \) represents a vector and \( R_i \), a scalar, denotes \( i \)th component of vector. Also, relevance vector generated at time \( t \) is denoted \( R_t \) with \( i \)th component denoted \( R_{t,i} \).

The objective is to learn a ranking function which ranks the documents associated with a query in such a way that more relevant documents are placed ahead of less relevant ones. The prevalent technique is to learn a scoring function and obtain a ranking by sorting the score vector in descending order. For \( X \in \mathcal{X} \), a linear scoring function is \( f_w(X) = X \cdot w = s^w \in \mathbb{R}^m \), where \( w \in \mathbb{R}^d \).

NDCG, cut off at \( k \leq m \) for a query with \( m \) documents, with relevance vector \( R \) and score vector \( s \) induced by a ranking function, is defined as follows:

\[
\text{NDCG}_k(s, R) = \frac{1}{Z_k(R)} \sum_{i=1}^k G(R_{\pi(i)}) D(i). \tag{1}
\]

Shorthand representation of \( \text{NDCG}_k(s, R) \) is \( \text{NDCG}_k \). Here, \( G(r) = 2^r - 1 \), \( D(i) = \frac{1}{\log_2(i+1)} \), \( Z_k(R) = \max_{\pi \in S_m} \sum_{i=1}^k G(R_{\pi(i)}) D(i) \). Further, \( S_m \) represents set of permutations over \( m \) objects.
\[\pi_s = \text{argsort}(s)\] is the permutation induced by sorting score vector \(s\) in descending order (we use \(\pi_s\) and \(\text{argsort}(s)\) interchangeably). A permutation \(\pi\) gives a mapping from ranks to documents and \(\pi^{-1}\) gives a mapping from documents to ranks. Thus, \(\pi(i) = j\) means document \(j\) is placed at position \(i\) while \(\pi^{-1}(i) = j\) means document \(i\) is placed at position \(j\). For \(k = m\), we denote \(\text{NDCG}_m(s, R)\) as \(\text{NDCG}(s, R)\). The popular performance measure, Average Precision (AP), is defined only for binary relevance vector, i.e., each component can only take values in \(\{0, 1\}\):

\[
\text{AP}(s, R) = \frac{1}{r} \sum_{j: R_{\pi_s(j)} = 1} \sum_{i < j} 1[R_{\pi_s(i)} = 1] / j
\]

where \(r = \|R\|_1\) is the total number of relevant documents.

All ranking performances measures are actually gains. When we say “NDCG induced loss”, we mean a loss function that simply subtracts NDCG from its maximum possible value, which is 1 (same for AP).

### 3. Perceptron for Classification

We will first briefly review the perceptron algorithm for classification, which executes online gradient descent (OGD) \cite{Zinkevich2003} on hinge loss during mistake rounds and achieves a bound on total number of mistakes. This will allow us to directly compare and contrast our extension of perceptron to the learning to rank setting. For full details, readers can refer to Sec. 3.3. in \cite{Shalev-Shwartz2011}.

In classification, an instance is of the form \(x \in \mathbb{R}^d\) and corresponding supervision (label) is \(y \in \{-1, 1\}\). A linear classifier is a scoring function \(g_w(\cdot)\), parameterized by \(w \in \mathbb{R}^d\), producing score \(g_w(x) = x \cdot w = s \in \mathbb{R}\). Classification of \(x\) is obtained by using “sign” predictor on \(s\), i.e., \(\text{sign}(s) \in \{-1, 1\}\). The loss is of the form: \(\ell(w, (x, y)) = 1[\text{sign}(x \cdot w) \neq y]\). The hinge loss is defined as: \(\phi(w, (x, y)) = [1 - y(x \cdot w)]_+\), where \([a]_+ = \max\{0, a\}\).

The perceptron algorithm operates on the loss \(f_t(w)\), defined on a sequence of data \(\{x_t, y_t\}_{t \geq 1}\), produced by an adaptive adversary as follows:

\[
f_t(w) = \begin{cases} 
[1 - y_t(x_t \cdot w)]_+ & \text{if } \ell(w_t, (x_t, y_t)) = 1 \\
0 & \text{if } \ell(w_t, (x_t, y_t)) = 0
\end{cases}
\]

It is important to understand the concept of the loss \(f_t(\cdot)\) and adaptive adversary here. The adversary knows the steps in the perceptron algorithm (Algorithm 1). Once the learner decides to play \(w_t\) at step \(t\), the adversary decides which function to play, either \([1 - y_t(x_t \cdot w)]_+\) or 0, depending on whether \(\ell(w_t, (x_t, y_t))\) is 1 or 0 respectively. Notice that \(f_t(w)\) is convex in both cases.

The perceptron updates a classifier \(g_{w_t}(\cdot)\) (effectively updates \(w_t\)) in an online fashion. The update occurs by application of OGD on the sequence of functions \(f_t(w)\) in the following way: perceptron initializes \(w_1 = \vec{0}\) and uses update rule \(w_{t+1} = w_t - \eta z_t\), where \(z_t \in \partial f_t(w_t)\) (\(z_t\) is a subgradient) and \(\eta\) is the learning rate. If \(\ell(w_t, (x_t, y_t)) = 0\), then \(f_t(w_t) = 0\); hence \(z_t = \vec{0}\). Otherwise, \(z_t = -y_t x_t \in \partial f_t(w_t)\). Thus,

\[
w_{t+1} = \begin{cases} 
w_t & \text{if } \ell(w_t, (x_t, y_t)) = 0 \\
w_t + \eta y_t x_t & \text{if } \ell(w_t, (x_t, y_t)) = 1
\end{cases}
\]

The perceptron algorithm for classification is described below:
Algorithm 1 Perceptron Algorithm for Classification

Learning rate $\eta > 0$, $w_1 = 0 \in \mathbb{R}^d$.
For $t = 1$ to $T$
    Receive $x_t$.
    Predict $p_t = \text{sign}(x_t \cdot w_t)$.
    Receive $y_t$
    If $\ell(w_t, (x_t, y_t)) \neq 0$
        $w_{t+1} = w_t + \eta y_t x_t$
    else
        $w_{t+1} = w_t$
End For

Theorem 1. Suppose that the perceptron for classification algorithm runs on an online sequence of data $\{(x_1, y_1), \ldots, (x_T, y_T)\}$ and let $R_x = \max_t \|x_t\|_2$. Let $f_t(\cdot)$ be defined as in Eq. 3. For all $u \in \mathbb{R}^d$, the perceptron mistake bound is:

$$
\sum_{t=1}^{T} \ell(w_t, (x_t, y_t)) \leq \sum_{t=1}^{T} f_t(u) + R_x \|u\|_2 \sqrt{\sum_{t=1}^{T} f_t(u) + R_x^2 \|u\|_2^2} \tag{5}
$$

In the special case where there exists $u$ s.t. $f_t(u) = 0$, $\forall t$, we have

$$
\forall T, \sum_{t=1}^{T} \ell(w_t, (x_t, y_t)) \leq R_x^2 \|u\|_2^2 \tag{6}
$$

As can be clearly seen from Eq. 5, the cumulative loss bound (i.e., total number of mistakes over $T$ rounds) is upper bounded by a cumulant of the function $f_t(\cdot)$. In the special case where there exists a perfect linear classifier with margin, Eq. 6 shows that the total number of mistakes is bounded, regardless of the number of instances.

One drawback of the bound in Eq. 6 is that the concept of margin is not explicit, i.e., it is hidden in the norm of the parameter of the perfect classifier ($\|u\|_2$). Let us assume that there is a linear classifier parameterized by a unit norm vector $u_\star$, such that all instances $x_t$ are not only correctly classified, but correctly classified with a margin $\gamma$, defined as:

$$
y_t(x_t \cdot u_\star) \geq \gamma, \forall T \tag{7}
$$

Then the following corollary holds:

Corollary 2. If the margin condition (7) holds, then total number of mistakes is upper bounded by $\frac{R_x^2}{\gamma^2}$, a bound independent of the number of instances in the online sequence.

4. A Novel Family of Listwise Surrogates

We define the novel SLAM family of loss functions: these are Surrogate, Large margin, Listwise and Lipschitz losses, Adaptable to multiple performance measures, and can handle Multiple graded
relevance. For score vector $s \in \mathbb{R}^m$, and relevance vector $R \in \mathcal{Y}$, the family of convex loss functions is defined as:

$$\phi^v_{\text{SLAM}}(s, R) = \min_{\delta \in \mathbb{R}^m} \sum_{i=1}^m v_i \delta_i$$

s.t. $\delta_i \geq 0, \forall i$, $s_i + \delta_i \geq \Delta + s_j$, if $R_i > R_j$, $\forall i, j$. \hspace{1cm} (8)

The constant $\Delta$ denotes margin and $v = (v_1, \ldots, v_m)$ is an element-wise non-negative weight vector. Different vectors $v$, to be defined later, yield different members of the SLAM family. Though $\Delta$ can be varied for empirical purposes, we fix $\Delta = 1$ for our analysis. The intuition behind the loss setting is that scores associated with more relevant documents should be higher, with a margin, than scores associated with less relevant documents. The weights decide how much weight to put on the errors.

The following reformulation of $\phi^v_{\text{SLAM}}(s, R)$ will be useful in later derivations.

$$\sum_{i=1}^m v_i \max(0, \max_{j=1,\ldots,m} \{1( R_i > R_j ) (1 + s_j - s_i) \} ) .$$ \hspace{1cm} (9)

**Lemma 3.** For any relevance vector $R$, the function $\phi^v_{\text{SLAM}}(\cdot, R)$ is convex.

**Proof.** It can be observed from Eq. 9. \hfill \Box

### 4.1 Weight Vectors Parameterizing the SLAM Family

As we stated after Eq. 8, different weight vectors lead to different members of the SLAM family. The weight vectors play a crucial role in the subsequent theoretical analysis. We will provide two weight vectors, $v^{\text{AP}}$ and $v^{\text{NDCG}}$, that result in upper bounds for AP and NDCG induced losses respectively. Later, we will discuss the necessity of choosing such weight vectors.

Since the losses in SLAM family is calculated with the knowledge of the relevance vector $R$, for ease of subsequent derivations, we can assume, without loss of generality, that documents are sorted according to their relevance levels. Thus, we assume that $R_1 \geq R_2 \geq \ldots \geq R_m$, where $R_i$ is the relevance of document $i$.

**Weight vector for AP loss:** Let $R \in \mathbb{R}^m$ be a binary relevance vector. Let $r$ be the number of relevant documents (thus, $R_1 = R_2 = \ldots = R_r = 1$ and $R_{r+1} = \ldots = R_m = 0$). We define vector $v^{\text{AP}} \in \mathbb{R}^m$ as

$$v_i^{\text{AP}} = \begin{cases} \frac{1}{r} & \text{if } i = 1, 2, \ldots, r \\ 0 & \text{if } i = r + 1, \ldots, m. \end{cases}$$ \hspace{1cm} (10)

**Weight vector for NDCG loss:** For a given relevance vector $R \in \mathbb{R}^m$, we define vector $v^{\text{NDCG}} \in \mathbb{R}^m$ as

$$v_i^{\text{NDCG}} = \frac{G(R_i) D(i)}{Z(R)}, \ i = 1, \ldots, m.$$ \hspace{1cm} (11)

**Note:** Both weights ensure that $v_1 \geq v_2 \geq \ldots \geq v_m$ (since $R_1 \geq R_2 \geq \ldots \geq R_m$). Using the weight vectors, we have the following upper bounds.
Theorem 4. Let $v^{AP} \in \mathbb{R}^m$ and $v^{NDCG} \in \mathbb{R}^m$ be the weight vectors as defined in Eq. (10) and Eq. (11) respectively. Let $AP(s, R)$ and $NDCG(s, R)$ be the AP value and NDCG value determined by relevance vector $R \in \mathbb{R}^m$ and score vector $s \in \mathbb{R}^m$. Then, the following inequalities hold, $\forall s, R$

$$\phi_{SLAM}^{AP}(s, R) \geq 1 - AP(s, R)$$

$$\phi_{SLAM}^{NDCG}(s, R) \geq 1 - NDCG(s, R).$$

The proof of the theorem is in Appendix A

4.2 Properties of SLAM Family and Upper Bounds

Listwise Nature of SLAM Family: The critical property for a surrogate to be considered listwise is that the loss must be calculated over the entire list of documents as a whole, with errors at the top penalized more than errors at the bottom. Since perfect ranking places the most relevant documents at top, errors corresponding to most relevant documents should be penalized more in SLAM in order to be considered a listwise family. Both $v^{NDCG}$ and $v^{AP}$ has the property that the more relevant documents get more weight.

Upper Bounds on NDCG and AP: By Theorem 4, the weight vectors make losses in SLAM family upper bounds on NDCG and AP induced losses. The SLAM loss family is analogous to the hinge loss in classification. Similar to hinge loss, the surrogate losses of SLAM family are 0 when the predicted scores respect the relevance labels (with some margin). The upper bound property will be crucial in deriving guarantees for a perceptron-like algorithm in learning to rank. Like hinge loss, the upper bounds can possibly be loose in some cases, but, as we show next, the upper bounding weights make SLAM family Lipschitz continuous with a small Lipschitz constant. This naturally restricts SLAM losses from growing too quickly. Empirically, we will show that the perceptron developed based on the SLAM family produce competitive performance on large scale industrial datasets. Along with the theory, the empirical performance supports the fact that upper bounds are quite meaningful.

Lipschitz Continuity of SLAM: Lipschitz continuity of an arbitrary loss, $\ell(s, R)$ w.r.t $s$ in $\ell_2$ norm, means that there is a constant $L_2$ such that $|\ell(s_1, R) - \ell(s_2, R)| \leq L_2 \|s_1 - s_2\|_2$, for all $s_1, s_2 \in \mathbb{R}^m$. By duality, it follows that $L_2 \geq \sup_s \|\nabla_s \phi(s, R)\|_2$. We calculate $L_2$ as follows:

Let $b_{ij} = \{1(R_i > R_j)(1 + s_j - s_i)\}$. The gradient of $\phi_{SLAM}^v$, w.r.t to $s$, from Eq. (9), is:

$$\nabla_s \phi_{SLAM}^v(s, R) = \sum_{i=1}^{m} v_i a^i,$$

where

$$a^i = \begin{cases} 0 \in \mathbb{R}^m & \text{if } \max_{j=1,...,m} b_{ij} \leq 0 \\ e_k - e_i \in \mathbb{R}^m & \text{otherwise} \\ k = \arg\max_{j=1,...,m} b_{ij} \end{cases}$$

and $e_i$ is a standard basis vector along coordinate $i$.

Since $\|a^i\|_1 \leq 2$, it is easy to see that $\|\nabla_s \phi_{SLAM}^v(s, R)\|_1 \leq 2 \sum_{i=1}^{m} v_i$. Since $\ell_1$ norm dominates $\ell_2$ norm, $\phi_{SLAM}^v(s, R)$ is Lipschitz continuous in $\ell_2$ norm whenever we can bound $\sum_{i=1}^{m} v_i$. It is easy to check that $\sum_{i=1}^{m} v_i^{AP} = 1$ and $\sum_{i=1}^{m} v_i^{NDCG} = 1$. Hence, $v^{NDCG}$ and $v^{AP}$ induce Lipschitz continuous surrogates, with Lipschitz constant at most 2.

Comparison with Surrogates Derived from Structured Prediction Framework: We briefly highlight the difference between SLAM and listwise surrogates obtained from the structured prediction framework [Chapelle et al. 2007, Yue et al. 2007, Chakrabarti et al. 2008]. Structured
prediction for ranking models assume that the supervision space is the space of full rankings of a document list. Usually a large number of full rankings are compatible with a relevance vector, in which case the relevance vector is arbitrarily mapped to a full ranking. In fact, here is a quote from one of the relevant papers [Chapelle et al., 2007], “It is often the case that this \( y_q \) is not unique and we simply take one of them at random” (\( y_q \) refers to a correct full ranking pertaining to query \( q \)). Thus, all but one correct full ranking will yield a loss. In contrast, in SLAM, documents with same relevance level are essentially exchangeable (observe Eq. (9)). Thus, all but one correct full ranking will yield a loss. In contrast, in SLAM, documents with same relevance level are essentially exchangeable (observe Eq. (9)). Thus, our assumption that documents are sorted according to relevance during design of weight vectors is without arbitrariness, and there will be no change in the amount of loss when documents within same relevance class are compared.

5. Perceptron-like Algorithms

We present a perceptron-like algorithm for learning a ranking function in an online setting, using the SLAM family. Since our proposed perceptron like algorithm works for both NDCG and AP induced losses, for derivation purposes, we denote a performance measure induced loss as Ranking-MeasureLoss (RML). Thus, RML can be NDCG induced loss or AP induced loss.

Informal Definition: The algorithm works as follows. At time \( t \), the learner maintains a linear ranking function, parameterized by \( w_t \). The learner receives \( X_t \), which is the document list retrieved for query \( q_t \) and ranks it. Then the ground truth relevance vector \( R_t \) is received and ranking function updated according to the perceptron rule.

Let \( b_{ij} = \{1 (R_i > R_j) (1 + s_j - s_i) \} \). For subsequent ease of derivations, we write SLAM loss from Eq. (9) as:

\[
\phi_{\text{SLAM}}^{w}(s^w, R) = \sum_{i=1}^{m} v_i c_i,
\]

where

\[
c_i = \begin{cases} 
0 & \text{if } \max_{j=1,...,m} b_{ij} \leq 0 \\
1 + s_k^w - s_i^w \in \mathbb{R} & \text{otherwise} \\
k = \arg\max_{j=1,...,m} b_{ij}.
\end{cases}
\] (14)

and \( s^w = X w \in \mathbb{R}^m \).

Like classification perceptron, our perceptron-like algorithm operates on the loss \( f_t(w) \), defined on a sequence of data \( \{X_t, R_t\}_{t \geq 1} \), produced by an adaptive adversary (i.e., an adversary who can see the learner’s move before making its move) as follows:

\[
f_t(w) = \begin{cases} 
\phi_{\text{SLAM}}^{w}(s^w_t, R_t) & \text{if } \text{RML}(s^w_t, R_t) \neq 0 \\
0 & \text{if } \text{RML}(s^w_t, R_t) = 0
\end{cases}
\] (15)

Here, \( s^w_t = X_t w \) and \( v_t = v_t^{\text{NDCG}} \) or \( v_t^{\text{AP}} \) depending on whether RML is NDCG or AP induced loss. Since weight vector \( v \) depends on relevance vector \( R \) (Eq. (10), (11)), the subscript \( t \) in \( v_t \) denotes the dependence on \( R_t \). Moreover, \( w_t \) is the parameter produced by our perceptron (Algorithm 2) at time \( t \), with the adaptive adversary being influenced by the move of perceptron (for similar, refer to Eq. (3) and discussion thereafter).

It is clear from Theorem 4 and Eq. (15) that \( f_t(w_t) \geq \text{RML}(s^w_t, R_t) \). It should also be noted that that \( f_t(\cdot) \) is convex in either of the two cases. Thus, we can run a variant of online gradient descent (OGD) algorithm [Zinkevich, 2003] to learn the sequence of parameters \( w_t \), starting with \( w_1 = 0 \). The OGD update rule, \( w_{t+1} = w_t - \eta z_t \), for some \( z_t \in \partial f_t(w_t) \) and step size \( \eta \), requires
a sub gradient $z_t$ that, in our case, is computed as follows. When $RML(s_t^{w_t}, R_t) = 0$, we have $z_t = 0 \in \mathbb{R}^d$. When $RML(s_t^{w_t}, R_t) \neq 0$, we have

$$z_t = X^T_t \left( \sum_{i=1}^{m} v_{t,i} a_{t,i} \right) \in \mathbb{R}^d,$$

(16)

where $e_k$ is the standard basis vector along coordinate $k$ and $c^t_i \in \mathbb{R}$ is as defined in Eq. (14) (with $s^w = s_t^{w_t} = X_t w_t$).

We now obtain a perceptron-like algorithm for the learning to rank problem.

**Algorithm 2 Perceptron Algorithm for Learning to Rank**

Learning rate $\eta > 0$, $w_1 = 0 \in \mathbb{R}^d$.

For $t = 1$ to $T$

- Receive $X_t$ (document list for query $q_t$).
- Set $s_t^{w_t} = X_t w_t$, predicted ranking output = $\text{argsort}(s_t^{w_t})$.
- Receive $R_t$
  - If $RML(s_t^{w_t}, R_t) \neq 0$ (Reminder: $RML(s_t^{w_t}, R_t) = RML(\text{argsort}(s_t^{w_t}), R_t)$)
    - $w_{t+1} = w_t - \eta z_t$ ($z_t$ from Eq. (16))
  - else
    - $w_{t+1} = w_t$

End For

5.1 Bound on Cumulative Loss

We provide a theoretical bound on the cumulative loss (as measured by RML) of perceptron for the learning to rank problem. The technique uses regret results of online convex optimization algorithms. We state the standard OGD bound used to get our main theorem [Zinkevich, 2003]. An important thing to remember is that OGD guarantee holds for convex functions played by an adaptive adversary, which is important for OGD based perceptron Algorithm

**Proposition (OGD regret).** Let $f_t$ be a sequence of convex functions. The update rule of function parameter is $w_{t+1} = w_t - \eta z_t$, where $z_t \in \partial f_t(w_t)$. Then for any $w \in \mathbb{R}^d$, the following regret bound holds after $T$ rounds,

$$\sum_{t=1}^{T} f_t(w_t) - \sum_{t=1}^{T} f_t(w) \leq \frac{||w||^2_2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} ||z_t||^2_2$$

(17)

We first control the norm of the subgradient $z_t$, defined in Eq. (16). To do this, we use the $p \to q$ norm of matrix.

**Definition (p → q norm).** Let $A \in \mathbb{R}^{m \times n}$ be a matrix. The $p \to q$ norm of $A$ is:

$$||A||_{p \to q} = \max_{v \neq 0} \frac{||Av||_q}{||v||_p}$$
Lemma 5. Let $R_X$ be the bound on the maximum $\ell_2$ norm of the feature vectors representing the documents. Let $v_{t,max} = \max_{i,j} \{ v_{i,j} \}, \forall i, j$ with $v_{t,i} > 0$, $v_{t,j} > 0$, and $m$ be bound on number of documents per query. Then we have the following $\ell_2$ norm bound,
$$\forall t, \|z_t\|_2^2 \leq 4 m R_X^2 v_{t,max} f_t(w_t).$$ (18)

Proof. For a mistake round $t$, we have $z_t = X_t^T (\sum_{i=1}^m v_{t,i} a_{t,i})$ from Eq. (16).

1st bound for $z_t$:
$$\|X_t^T (\sum_{i=1}^m v_{t,i} a_{t,i})\|_2 \leq \|X_t^T\|_{1 \rightarrow 2} \|\sum_{i=1}^m v_{t,i} a_{t,i}\|_1 \leq 2R_X \sum_{i=1}^m v_{t,i} \leq 2R_X.$$ The first inequality uses the $1 \rightarrow 2$ norm and last inequality holds because $\sum_{i=1}^m v_i^{\nu_{DGC}} = 1$ and $\sum_{i=1}^m v_i^{AP} = 1$.

2nd bound for $z_t$ (exploiting self-bounding property of SLAM):
We note that in a mistake round, $RML(s_{t,i}^{w_t}, R_t) \neq 0$. Thus, $\exists i', k'$ s.t $R_{t,i'} > R_{t,k'}$ but $s_{t,i'}^{w_t} < s_{t,k'}^{w_t}$ (this is the condition of at least 1 pair of documents whose ranks are inconsistent with their relevance levels.) Now, $\phi_{SLAM}^{v_t}(s_t^{w_t}, R_t) = \sum v_{t,i} c_t^i$ (Eq. (14)). For $(i', k')$, we have $c_t^i \geq 1 + s_{t,k}^{w_t} - s_{t,i}^{w_t} > 1$.

Since $R_{t,i'} > R_{t,k'}$, document $i'$ has strictly greater than minimum possible relevance, i.e, $R_{t,i'} > 0$. By our calculations of weight vector $v$ for both NDCG and AP, we have $v_{t,i'} > 0$. Thus, by definition, $v_{t,max} \geq 1$ (since $v_{t,i'} > 0$ and $\frac{v_{t,i'}}{v_{t,j}} = 1$ and $v_{t,max} = \max_{i,j} \{ \frac{v_{t,i}}{v_{t,j}} \}, \forall i, j$ with $v_{t,i} > 0, v_{t,j} > 0$).

Then, $\forall i, v_{t,i} \leq v_{t,max} \cdot v_{t,i'} \leq v_{t,max} \cdot v_{t,i'} \cdot c_t^i$. Thus, we have: $\sum_{i=1}^m v_{t,i} \leq m v_{t,max} v_{t,i'} c_t^i \leq m v_{t,max} (\sum_{i=1}^m v_{t,i} c_t^i) = m v_{t,max} \phi_{SLAM}^{v_t}(s_t^{w_t}, R_t)$.

Thus, $\|z_t\|_2 \leq 2R_X \sum_{t,i} \leq 2R_X m v_{t,max} \phi_{SLAM}^{v_t}(s_t^{w_t}, R_t)$.

Combining 1st and 2nd bound for $z_t$, we get $\|z_t\|_2 \leq 2R_X m v_{t,max} \phi_{SLAM}^{v_t}(s_t^{w_t}, R_t)$, for mistake rounds.

Since, for non-mistake rounds, we have $z_t = 0$ and $f_t(w_t) = 0$, we get the final inequality.

□

Taking $\max_{t=1}^T v_{t,max} \leq v_{max}$, we have the following theorem, which uses the norm bound on $z_t$:

Theorem 6. Suppose Algorithm \ref{alg:ams} receives a sequence of instances $(X_1, R_1), \ldots, (X_T, R_T)$ and let $R_X$ be the bound on the maximum $\ell_2$ norm of the feature vectors representing the documents. Then the following inequality holds, after optimizing over learning rate $\eta, \forall w \in \mathbb{R}^d$:
$$\sum_{t=1}^T RML(s_t^{w_t}, R_t) \leq \sum_{t=1}^T f_t(w) + \sqrt{4\|w\|^2 m R_X^2} \sqrt{\sum_{t=1}^T f_t(w) + 4\|w\|^2 m R_X^2 v_{max}}. \quad (19)$$
In the special case where there exists \( w \) s.t. \( f_t(w) = 0, \forall t \), we have

\[
\sum_{i=1}^{T} RML(s_{t}^{w_t}, R_t) \leq 4\|w\|_2^2 m R_X^2 v_{\text{max}}. \tag{20}
\]

**Proof.** The proof follows by plugging in expression for \( \|z_t\|^2 \) (Lemma 5) in OGD equation (Prop. OGD Regret), optimizing over \( \eta \), using the algebraic trick: \( x - b\sqrt{x} - c \leq 0 \implies x \leq b^2 + c + b\sqrt{c} \) and then using the inequality \( f_t(w_t) \geq \phi_{t}^{SLAM}(s_{t}^{w_t}, R_t) \).

**Note:** The perceptron bound, in Eq. [19] is a loss bound, i.e., the left hand side is cumulative NDCG/AP induced loss while right side is function of cumulative surrogate loss. We discuss in details the significance of this bound later.

Like perceptron for binary classification, the constant in Eq. [19] needs to be expressed in terms of a “margin”. A natural definition of margin in case of ranking data is as follows: let us assume that there is a linear scoring function parameterized by a unit norm vector \( w^* \), such that all documents for all queries are ranked not only correctly, but correctly with a margin \( \gamma \):

\[
\min_{i,j: R_{t,i} > R_{t,j}} w^\top X_{t,i} - w^\top X_{t,j} \geq \gamma. \tag{21}
\]

**Corollary 7.** If the margin condition (21) holds, then total loss, for both NDCG and AP induced loss, is upper bounded by \( \frac{4mR_X^2 v_{\text{max}}^2}{\gamma^2} \), a bound independent of the number of instances in the online sequence.

**Proof.** Fix a \( t \) and the example \((X_t, R_t)\). Set \( w = w^* / \gamma \). For this \( w \), we have

\[
\min_{i,j: R_{t,i} > R_{t,j}} w^\top X_{t,i} - w^\top X_{t,j} > 1,
\]

which means that

\[
\min_{i,j: R_{t,i} > R_{t,j}} s_{t,i}^w - s_{t,j}^w > 1
\]

This immediately implies that \( 1(R_{t,i} > R_{t,j})(1+s_{t,i}^w-s_{t,j}^w) \leq 0, \forall i, j \). Therefore, \( \phi_{t}^{SLAM}(s_{t}^{w_t}, R_t) = 0 \) and hence \( f_t(w) = 0 \). Since this holds for all \( t \), we have \( \sum_{t=1}^{T} f_t(w) = 0. \)

**5.1.1 Perceptron Bound-General Discussion**

We remind once again that \( RML(s_{t}^{w_t}, R_t) \) is either \( 1 - \text{NDCG}(s_{t}^{w_t}, R_t) \) or \( 1 - \text{AP}(s_{t}^{w_t}, R_t) \), depending on measure of interest.

**Dependence of perceptron bound on number of documents per query:** The perceptron bound in Eq. [19] is meaningful only if \( v_{\text{max}} \) is a finite quantity.

For AP, it can be seen from the definition of \( v^{\text{AP}} \) in Eq. [10] that \( v_{\text{max}} = 1 \). Thus, for AP induced loss, the constant in the perceptron bound is: \( \frac{4mR_X^2}{\gamma^2} \).

For NDCG, \( v_{\text{max}} \) depends on maximum relevance level. Assuming maximum relevance level is finite (in practice, maximum relevance level is usually below 5), \( v_{\text{max}} = O(\log(m)) \). Thus, for
NDCG induced loss, the constant in the perceptron bound is: \( \frac{4m \log(m) R_x^2}{\gamma^2} \).

**Significance of perceptron bound:** The main perceptron bound is reflected in Eq. [19] with the special case being captured in Corrollary. [7] At first glance, the bound might seem non-informative because the left side is the cumulative NDCG/AP induced loss bound, while the right side is a function of the cumulative surrogate loss.

The first thing to note is that the perceptron bound is derived from the regret bound in Eq. [17], which is the famous regret bound of the OGD algorithm applied to an arbitrary convex, Lipschitz surrogate. So, even ignoring the bound in Eq. [19], the perceptron algorithm is a valid online algorithm, applied to the sequence of convex functions \( f_t(\cdot) \), to learn ranking function \( w_t \), with a meaningful regret bound. Second, as we had mentioned in the introduction, our perceptron bound is the extension of perceptron bound in classification, to the cumulative NDCG/AP induced losses in the learning to rank setting. This can be observed by noticing the similarity between Eq. [19] and Eq. [5]. In both cases, the cumulative target loss on the left is bounded by a function of the cumulative surrogate loss on the right, where the surrogate is the hinge (and hinge like SLAM) loss. The interesting aspects of perceptron loss bound becomes apparent on close investigation of the cumulative surrogate loss term \( \sum_{t=1}^{T} f_t(w) \) and comparing with the regret bound. It is well known that when OGD is run on any convex, Lipschitz surrogate, the guarantee on the regret scales at the rate \( O(\sqrt{T}) \). So, if we only ran OGD on an arbitrary convex, Lipschitz surrogate, then, even with the assumption of existence of a perfect ranker, the upper bound on the cumulative loss would have scaled as \( O(\sqrt{T}) \). However, in the perceptron loss bound, if \( \sum_{t=1}^{T} f_t(w) = o(T^{\alpha}) \), then the upper bound on the cumulative loss would scale as \( O(T^{\alpha}) \), which can be much better than \( O(T^{1/2}) \) for \( \alpha < 1/2 \). In the best case of \( \sum_{t=1}^{T} f_t(w) = 0 \), the total cumulative loss would be bounded, irrespective of the number of instances.

**Comparison and contrast with perceptron for classification:** The perceptron for learning to rank is an extension of the perceptron for classification, both in terms of the algorithm and the loss bound. To obtain the perceptron loss bounds in the learning to rank setting, we had to address multiple non-trivial issues, which do not arise in the classification setting. Among them, perhaps, the most challenging was that, unlike in classification, the NDCG/AP losses are not \( \{0, 1\} \)-valued. The analysis is trivial in classification perceptron since on a mistake round, the absolute value of gradient of hinge loss is 1, which is same as the loss itself. In our setting, Lemma [5] is crucial, where we exploit the structure of SLAM surrogate to bound the square of gradient by the surrogate loss.

5.1.2 **PERCEPTRON BOUND DEPENDENT ON NDCG CUT-OFF POINT**

The bound on the cumulative loss in Eq. [19] is dependent on \( m \), the maximum number of documents per query. It is often the case in learning to rank that though a list has \( m \) documents, the focus during evaluation is on the top \( k \) ranked documents (\( k \ll m \)). The measure used for top-\( k \) documents is NDCG\(_k\) (Eq. [1]) (there does not exist an equivalent definition for AP).

We consider a modified set of weights \( v^{\text{NDCG}_k} \) s.t. \( \phi^{\text{NDCG}_k}(s, R) \geq 1 - \text{NDCG}_k(s, R) \) holds \( \forall s \), for every \( R \). We provide the definition of \( v^{\text{NDCG}_k} \) later in the proof of Theorem [8].

Overloading notation with \( v_t = v^{\text{NDCG}_k}_t \), let \( v_{t,\max} = \max_{i,j} \left\{ \frac{v_{t,i}}{v_{t,j}} \right\} \) with \( v_{t,i} > 0, v_{t,j} > 0 \) and \( v_{\max} = \max_{t=1}^{T} v_{t,\max} \).
Theorem 8. Suppose the perceptron algorithm receives a sequence of instances \((X_1, R_1), \ldots, (X_T, R_T)\). Let \(k\) be the cut-off point of NDCG. Also, for any \(w \in \mathbb{R}^d\), let \(f_t(w)\) be as defined in Eq. (15), but with \(\phi^{NDCG}_{s, R_t}(s^w, R_t) = \phi^{SLAM}_{s, R_t}(s^w, R_t)\). Then, the following inequality holds, after optimizing over learning rate \(\eta\),

\[
\sum_{t=1}^{T} (1 - NDCG_k(s^w_t, R_t)) \leq \sum_{t=1}^{T} f_t(w) + \sqrt{4\|w\|^2 kR_X^2 v_{\max}} \sum_{t=1}^{T} f_t(w) + 4\|w\|^2 kR_X^2 v_{\max}
\]

In the special case where there exists \(w\) s.t. \(f_t(w) = 0\), \(\forall t\), we have

\[
\sum_{t=1}^{T} (1 - NDCG_k(s^w_t, R_t)) \leq 4\|w\|^2 kR_X^2 v_{\max}.
\]

**Discussion:** Assuming maximum relevance level is finite, we have \(v_{\max} = O(\log(k))\) (using definition of \(v^{NDCG}\)). Thus, the constant term in the perceptron bound for NDCG\(_k\) induced loss is: \(4\|w\|^2 k\log(k)R_X^2\). This is a significant improvement from original error term, even though the perceptron algorithm is running on queries with \(m\) documents, which can be very large. Margin dependent bound can be defined in same way as before.

**Proof.** We remind again that ranking performance measures only depend on the permutation of documents and individual relevance level. They do not depend on the identity of the documents. Documents with same relevance level can be considered to be interchangeable, i.e, relevance levels create equivalence classes. Thus, w.l.o.g, we assume that \(R_1 \geq R_2 \geq \ldots \geq R_m\) and documents with same relevance level are sorted according to score. Also, \(\pi^{-1}(i)\) means position of document \(i\) in permutation \(\pi\).

We define \(v^{NDCG}_k\) as

\[
v^{NDCG}_i = \begin{cases} 
\frac{G(R_i)D(i)}{Z_k(R)} & \text{if } i = 1, 2, \ldots, k \\
0 & \text{if } i = k + 1, \ldots, m.
\end{cases}
\]

We now prove the upper bound property that \(\phi^{SLAM}_s(R) \geq 1 - \text{NDCG}_k(s, R)\) holds \(\forall s\), for every \(R\). We have the following equations:

\[
\sum_{i=1}^{m} \frac{G(R_i)D(i)1(i \leq k)}{Z_k(R)} = 1 \text{ and } \text{NDCG}_k(s, R) = \sum_{i=1}^{m} \frac{G(R_i)D(\pi^{-1}_s(i))1(\pi^{-1}_s(i) \leq k)}{Z_k(R)}.
\]

\[
\implies 1 - \text{NDCG}_k(s, R) = \sum_{i=1}^{m} \frac{G(R_i)(D(i)1(i \leq k) - D(\pi^{-1}_s(i))1(\pi^{-1}_s(i) \leq k))}{Z_k(R)}.
\]

For \(i > k\): \(D(i)1(i \leq k) = 0\) and since \(D(\pi^{-1}_s(i))\) is non-negative, every term in \(1 - \text{NDCG}_k(s, R)\) is non-positive for \(i > k\).

For \(i \leq k\), there are four possible cases:

1. \(i \geq \pi^{-1}_s(i)\) and \(\pi^{-1}_s(i) > k\). This is infeasible since \(i \leq k\).
2. \( i \geq \pi_s^{-1}(i) \) and \( \pi_s^{-1}(i) \leq k \). In this case, the numerator in \( 1 - \text{NDCG}_k \) is \( G(R_i)(D(i) - D(\pi_s^{-1}(i))) \). Now, since \( D(\cdot) \) is a decreasing function, the contribution of the document \( i \) to NDCG induced loss is non-positive and can be ignored (since SLAM by definition is sum of positive weighted indicator functions).

3. \( i < \pi_s^{-1}(i) \) and \( \pi_s^{-1}(i) > k \). In this case, the numerator in \( 1 - \text{NDCG}_k \) is \( G(R_i)D(i) \). Since \( i < \pi_s^{-1}(i) \), that means document \( i \) was outscored by a document \( j \), where \( i < j \) (otherwise, document \( i \) would have been put in a position same or above what it is at currently, by \( \pi_s \), i.e., \( i \geq \pi_s^{-1}(i) \).) Moreover, \( R_i > R_j \) (because of the assumption that within same relevance class, scores are sorted). Hence the indicator of SLAM at \( i \) would have come on and \( v_{i}^{\text{NDCG}_k} = \frac{G(R_i)(D(i) - D(\pi_s^{-1}(i)))}{Z_k(R)} \).

4. \( i < \pi_s^{-1}(i) \) and \( \pi_s^{-1}(i) \leq k \). In this case, the numerator in \( 1 - \text{NDCG}_k \) is \( G(R_i)(D(i) - D(\pi_s^{-1}(i))) \). By same reason as c.), the indicator of SLAM at \( i \) would have come on and \( v_{i}^{\text{NDCG}_k} > \frac{G(R_i)(D(i) - D(\pi_s^{-1}(i)))}{Z_k(R)} \) by definition of \( v_{i}^{\text{NDCG}_k} \) and the fact that \( D(i) > D(\pi_s^{-1}(i)) \).

Hence, the upper bound property holds.

The proof of Theorem 8 now follows directly following the argument in the proof of Lemma 5 by noting a few things:

a) \( \sum_{i=1}^{k} v_{i}^{\text{NDCG}_k} = 1 \). b) \( \phi_{\text{SLAM}}(s, R) \) has same structure as \( \phi_{\text{SLAM}}(s, R) \) but with different weights. Hence structure of \( z_i \) remains same but with weights of \( v_{i}^{\text{NDCG}_k} \).

Hence, 1st bound on gradient of \( z_i \) in proof of Lemma 5 remains same. For the 2nd bound on gradient of \( z_i \), the crucial thing that changes is that \( \sum_{i=1}^{m} v_{i}^{t} \leq k v_{t, \text{max}} v_{t, i} e_i \), with the new definitions of \( v_{t, \text{max}} \) according to \( v_{i}^{\text{NDCG}_k} \). This implies \( 2R_X \sum v_{t, i} \leq 2R_X k v_{t, \text{max}} \phi v_{t}(s_{i}^{w_{t}}, R_t) \).

6. Minimax Bound on Cumulative NDCG/ AP Induced Loss

6.1 Lower Bound

The following theorem gives a lower bound on the cumulative NDCG/AP induced loss, achievable by any deterministic online Algorithm

**Theorem 9.** Suppose the number of documents per query is restricted to two and relevance vectors are restricted to being binary graded. Let \( X = \{X \in \mathbb{R}^{2 \times d} \mid \|X_i\|_2 \leq R_X \} \) and \( \frac{R_X}{\gamma^2} \leq d \). Then, for any deterministic online algorithm, there exists a ranking dataset which is separable by margin \( \gamma \) (Eq. 21), on which the algorithm suffers at least \( \left\lfloor \frac{R_X^2}{\gamma^2} \right\rfloor \) cumulative NDCG/AP induced loss.

**Proof.** Let \( T = \left\lfloor \frac{R_X^2}{\gamma^2} \right\rfloor \). Since \( \frac{R_X^2}{\gamma^2} \leq d \), hence \( T \leq d \) and \( T \gamma^2 \leq R_X^2 \). Let a ranking dataset consist of the following \( T \) document matrices: document feature matrix \( X_i = [R_X \cdot e_i; -R_X \cdot e_i]^\top \), where \( e_i \) is the unit vector of length \( d \) with 1 in \( i \)th coordinate and 0 in others.

Given an algorithm \( A \), let the relevance vectors for the dataset be as following: for matrix \( X_i \), if \( A \) ranks 1st document above 2nd, then \( R_{i,2} = 1, R_{i,1} = 0 \); else, the relevances are reversed.

14
For the above dataset, \( \mathcal{A} \) will make a ranking mistake in each round and NDCG/AP induced loss will be \( O(T) = O\left(\frac{R^2}{\gamma^2}\right) \).

Now, it remains to be shown that the dataset is actually separable with margin \( \gamma \) by a ranking function with unit norm parameter.

Let a ranking function parameter \( w \in \mathbb{R}^d \) be defined as follows: \( \forall i \in [T] \), the \( i \)th component of \( w \) is

\[
\begin{align*}
  w_i = \begin{cases} 
  \frac{\gamma}{2} \cdot R_{X} & \text{if } R_{i,1} > R_{i,2} \\
  \frac{-\gamma}{2} \cdot R_{X} & \text{otherwise}
  \end{cases}
\end{align*}
\]

For \( i \not\in [T] \) but \( i \in [d] \), \( w_i = 0 \).

The unit norm condition holds because \( \|w\|^2 = \frac{T^2 \gamma^2}{4 R_X^2} \leq 1 \).

Let \( X_{i,j} \) indicate \( j \)th row of matrix \( X_i \). The margin condition holds as follows: \( \forall i \in [T] \), if \( R_{i,1} > R_{i,2} \), then \( X_{i,1} \cdot w - X_{i,2} \cdot w = \frac{\gamma}{2} - \frac{-\gamma}{2} = \gamma \), else if \( R_{i,1} < R_{i,2} \), then \( X_{i,2} \cdot w - X_{i,1} \cdot w = \frac{\gamma}{2} - \frac{-\gamma}{2} = \gamma \).

6.2 Algorithm Achieving Lower Bound

We will show that the lower bound established in the previous section is actually the minimax bound, achievable by another perceptron type algorithm. Thus, Algorithm 2 is sub-optimal in terms of the bound achieved, since it has a dependence on number of documents per query.

Our algorithm is inspired by the work of Crammer and Singer [2002]. Following their work, we define a new constrained optimization problem for ranking as follows:

\[
\phi_C(s, R) = \min \delta \quad \text{s.t.} \quad \delta \geq 0, \quad s_i + \delta \geq \Delta + s_j, \quad \text{if} \quad R_i > R_j, \quad \forall i, j.
\] (25)

The above constrained optimization problem can be recast as a hinge-like convex surrogate:

\[
\phi_C(s, R) = \max_{i \in [m]} \max_{j \in [m]} 1 \left[R(i) > R(j)\right] (1 + s_j - s_i)_+ \] (26)

The key difference between the above surrogate and the previously proposed SLAM family of surrogates is that the above surrogate does not adapt to different ranking measures and does not exhibit the listwise property (i.e., it does not differentiate between errors at top or bottom of rank list).

Similar to Algorithm 2, we define a sequence of losses \( f_t(w) \), defined on a sequence of data \( \{X_t, R_t\}_{t \geq 1} \), as follows:

\[
f_t(w) = \begin{cases} 
  \phi_C(s^w_t, R_t) & \text{if } \text{RML}(s^w_t, R_t) \neq 0 \\
  0 & \text{if } \text{RML}(s^w_t, R_t) = 0
  \end{cases}
\] (27)

Here, \( s^w_t = X_t w \) and \( w_t \) is the parameter produced by Algorithm 3 at time \( t \), with the adaptive adversary being influenced by the move of perceptron. Note that \( f_t(w_t) \geq \text{RML}(s^w_t, R_t) \), since, \( f_t(w) \) is always non-negative and if \( \text{RML}(s^w_t, R_t) > 0 \), there is at least one pair of documents whose scores do not agree with their relevances. At that point, the surrogate value becomes greater than 1.
During a mistake round, the gradient $z$ is calculated as follows: let $i^*, j^*$ be the indices that achieve the max in Eq 26. Then,

$$z = \nabla_w \phi_C(s^w, R) = X^T \{(-e_{i^*} + e_{j^*})1[R(i^*) > R(j^*)]1[1 + s_{j^*} - s_{i^*} \geq 0]\}. \quad (28)$$

Note that if there are multiple index pairs achieving the max, then the subgradient can be written as a convex combination of subgradients at each of the pairs.

**Algorithm 3** New Perceptron Algorithm Achieving Lower Bound

Learning rate $\eta > 0$, $w_1 = 0 \in \mathbb{R}^d$.

For $t = 1$ to $T$

Receive $X_t$ (document list for query $q_t$).

Set $s^{w_t}_t = X_tw_t$, predicted ranking output = $\text{argsort}(s^{w_t}_t)$.

Receive $R_t$

If $\text{RML}(s^{w_t}_t, R_t) \neq 0$ (Reminder: $\text{RML}(s^{w_t}_t, R_t) = \text{RML}(\text{argsort}(s^{w_t}_t), R_t)$)

$$w_{t+1} = w_t - \eta z_t \quad \text{(z_t from Eq. (28))}$$

else

$$w_{t+1} = w_t$$

End For

We have the following result:

**Theorem 10.** Suppose Algorithm 3 receives a sequence of instances $(X_1, R_1), \ldots, (X_T, R_T)$. Let $R_X$ be the bound on the maximum $\ell_2$ norm of the feature vectors representing the documents and $f_t(w)$ be as defined in Eq 27. Then the following inequality holds, after optimizing over learning rate $\eta$, $\forall w \in \mathbb{R}^d$:

$$\sum_{t=1}^{T} \text{RML}(s^{w_t}_t, R_t) \leq \sum_{t=1}^{T} f_t(w) + 2\|w\| R_X \sqrt{\sum_{t=1}^{T} f_t(w) + 4\|w\|^2 R_X^2}. \quad (29)$$

In the special case where there exists $w$ s.t. $f_t(w) = 0$, $\forall t$, we have

$$\sum_{t=1}^{T} \text{RML}(s^{w_t}_t, R_t) \leq 4\|w\|^2 R_X^2. \quad (30)$$

**Proof.** We first bound the $\ell_2$ norm of the gradient.

From Eq 28 we have:

**1st bound for $z_t$:**

$$\|z_t\|_2 \leq \|X^T\|_1 \|(-e_{i^*} + e_{j^*})1[R(i^*) > R(j^*)]1[1 + s_{j^*} - s_{i^*} \geq 0]\|_1 \leq 2R_X.$$ 

**2nd bound for $z_t$:**

On a mistake round, there exists at least 1 pair of documents, whose scores and relevance levels are discordant. Thus, $\phi_C(s^w, R) > 1$, and hence, $\|z_t\|_2 \leq 2R_X \leq 2R_X \phi_C(s^{w_t}_t, R_t)$.

Thus, $\|z_t\|_2 \leq 4R_X^2 \phi_C(s^{w_t}_t, R_t)$. Since $\|z_t\| = 0$ on non-mistake round, we finally have:
\[ \|z_t\|_2^2 \leq 4R^2 X f_t(w_t), \forall t. \]

The proof then follows as previous: by plugging in expression for \( \|z_t\|_2^2 \) in OGD equation (Prop. OGD Regret), optimizing over \( \eta \), using the algebraic trick: \( x - b\sqrt{x} - c \leq 0 \implies x \leq b^2 + c + b\sqrt{c} \) and then using the inequality \( f_t(w_t) \geq \text{RML}(s_{w_t}^1, R_t) \).

We have similar margin based bound.

**Corollary 11.** If the margin condition (21) holds, then total loss, for both NDCG and AP induced loss, is upper bounded by \( \frac{4R^2}{\gamma^2} \), a bound independent of the number of instances in the online sequence.

**Proof.** Proof is similar to Corollary 7.

**Comparison of Algorithm 2 and Algorithm 3** Both of our proposed perceptron type algorithms fill the “classification analogue” gap in the learning to rank literature. Algorithm 3 achieves the lower bound on separable datasets, unlike Algorithm 2 which scales with number of documents per query. However, Algorithm 3 operates on a surrogate (Eq 26) which is not listwise in nature, even though it forms an upper bound on the listwise ranking measures. To emphasize, the surrogate does not differentiably weigh between errors at different points of the ranked list, which is an important property of popular surrogates in learning to rank. As our empirical results show (Sec. 8), on commercial datasets which are not separable, Algorithm 3 has significantly worse performance than Algorithm 2.

**7. Related Work in Perceptron for Ranking**

There exists a number of papers in the literature dealing with perceptron in the context of ranking. We will compare and contrast our work with existing work, paying special attention to the papers which come closest to our model.

First, we would like to point out that, to the best of our knowledge, there is no paper which establishes document independent bound for NDCG, cut-off at the top \( k \) position (Theorem 8). Moreover, we believe our work, for the first time, formally establishes minimax bound , achievable by any deterministic online algorithm, in the learning to rank setting, under assumption of separability.

[Crammer and Singer 2001] was one of the first papers to introduce perceptron in ranking. The setting as well as results of Crammer’s perceptron is quite different from ours. The paper assumes there is a fixed set of ranks \( \{1, 2, \ldots, k\} \). An instance is a vector of the form \( x \in \mathbb{R}^d \) and the supervision is one of the \( k \) ranks. The perceptron has to learn the correct ranking of \( x \), with the loss being 1 if correct rank is not predicted. The paper does not deal with query-documents list and does not consider learning to rank measures like NDCG/AP.

Another paper whose results have some similarity to ours is [Wang et al. 2015]. The paper introduces algorithms for online learning to rank, but does not claim to have any “perceptron type” results. However, their main theorem (Theorem 2) has a perceptron bound flavor to it, where the cumulative NDCG/AP losses are upper bounded by cumulative surrogate loss and a constant. The major differences with our results are these: [Wang et al. 2015] has a different instance/supervision setting and consequently has different surrogate loss. It is assumed that for each query \( q \), only a
pair of documents \((x_i, x_j)\) are received at each online round, with the supervision being \(\{+1, -1\}\), depending on whether \(x_i\) is more/less relevant than \(x_j\). The surrogate loss is defined at pair of documents level, and not at a query-document matrix level. Moreover, there is no equivalent result to our Theorem 8, neither is any kind of minimax bound established.

A third paper which has similarity to ours is Jain et al. [2015]. Since the predtron algorithm is more general, the bound achieved by predtron, applied to the ranking case, has a scaling factor \(O(m^5)\), significantly worse than our linear scaling. Moreover, it does not have the NDCG\(_k\) scaling as \(k\) proved anywhere.

There are other, less related papers; all of which deal with perceptron in ranking, in some form or the other. The paper Ni and Huang [2008] has similar setting to Jain et al. [2015], but introduces the concept of margin in that particular setting, with corresponding perceptron bounds. The paper does not deal with query-document matrices, nor NDCG/AP induced losses. Another couple of papers, Elsas et al. [2008] and Harrington [2003] introduce online perceptron based ranking algorithms, but have no established theoretical results/bounds. Another paper, Shen and Joshi [2005] has a perceptron type algorithm with theoretical guarantee, but in their paper, the supervision is in form of full rankings (instead of relevance vectors).

8. Experiments

We conducted experiments on a simulated dataset and three large scale commercial datasets, mainly to validate our theoretical results. Our results demonstrate the following:

- We simulated a margin \(\gamma\) separable dataset. On that dataset, the two algorithms (Algorithm 2 and Algorithm 3) ranks all but a finite number of instances correctly, validating that our algorithms have perceptron like property.

- On three commercial datasets, which are not separable, Algorithm 2 shows competitive performance to the baseline algorithm, while Algorithm 3 performs quite poorly on two of the datasets. The results support the notion that the SLAM listwise surrogates, which differentially penalize errors depending on position in the ranked list, is practically more useful than the surrogate in Eq 26 though the latter has tighter loss bound on separable datasets.

**Baseline Algorithm:** We compared our algorithms with the online version of the popular ListNet ranking algorithm [Cao et al., 2007]. ListNet is not only one of the most cited ranking algorithms (over 700 citations according to Google Scholar), but also one of the most validated algorithms [Tax et al., 2015]. We conducted online gradient descent on the cross-entropy convex surrogate of ListNet to learn a ranking function in an online manner. While there exists ranking algorithms which have demonstrated better empirical performance than ListNet, they are generally based on non-convex surrogates with non-linear ranking functions. These algorithms cannot be converted in a straight forward way (or not at all) into online algorithms which learn from streaming data. We also did not compare our algorithms with other perceptron algorithms since they do not usually have similar setting to ours and would require substantial modifications for direct comparison.

**Experimental Setting:** For all datasets, we report average NDCG\(_{10}\) over a time horizon. Average NDCG\(_{10}\) at iteration \(t\) is the cumulative NDCG\(_{10}\) up to iteration \(t\), divided by \(t\). We remind that at each iteration \(t\), a document matrix is ranked by the algorithm, with the performance (according to
NDCG$_{10}$) measured against the true relevance vector corresponding to the document matrix. For all the algorithms, the corresponding best learning rate $\eta$ was fixed after conducting experiments with multiple different rates and observing the best time averaged NDCG$_{10}$ over a fixed time interval. We did not conduct any experiments with AP since the real datasets have multi-graded relevance and NDCG is a suitable measure for such datasets.

**Simulated Dataset:** We simulated a margin separable dataset (Eq. (21)). Each query had $m = 20$ documents, each document represented by 20 dimensional feature vector, and five different relevance level \{4, 3, 2, 1, 0\}, with relevances distributed uniformly over the documents. The feature vectors of equivalent documents (i.e., documents with same relevance level) were generated from a Gaussian distribution, with documents of different relevance levels generated from different Gaussian distribution. A 20 dimensional unit norm ranker was generated from a Gaussian distribution, which induced separability with margin. Fig. 1 compares performance of Algorithm 2, Algorithm 3 and online ListNet. The NDCG$_{10}$ values of the perceptron type algorithms rapidly converge to 1, validating their perceptron type property. To re-iterate, since for separable datasets, cumulative NDCG induced loss is bounded by constant, hence, the time averaged NDCG should rapidly converge to 1. The OGD algorithm for ListNet has only a regret guarantee of $O(\sqrt{t})$; hence the time averaged regret converges at rate $O\left(\frac{1}{\sqrt{t}}\right)$, i.e., its convergence is significantly slower than the perceptron type algorithms.

![Figure 1: Time averaged NDCG$_{10}$ for Algorithm 2, Algorithm 3, and ListNet, for separable dataset. The two perceptron type algorithms have imperceptible difference.](image-url)

**Commercial Datasets:** We chose three large scale industry published ranking datasets to analyze the performance of our algorithms. MSLR-WEB10K [Liu et al., 2007] is the dataset published by Microsoft’s Bing team, consisting of 10,000 unique queries, with feature dimension of size 245 and 5 distinct relevance levels. Yahoo Learning to Rank Challenge dataset [Chapelle and Chang, 2011] consists of 19,944 unique queries, with feature dimension of size 700 and 5 distinct relevance levels. Yandex, Russia’s biggest search engine, published a dataset (link to the dataset given in the work of Chapelle and Chang [2011]) consisting of 9124 queries, with feature dimension of size 245 and 5 distinct relevance levels. Algorithm 2 performs slightly better than ListNet on MSLR-WEB dataset (average NDCG@10 over last ten iterations= 0.25 vs 0.22), performs slightly worse
Figure 2: Time averaged NDCG\textsubscript{10} for Algorithm 2, Algorithm 3, and ListNet for 3 commercial datasets. While Algorithm 2 has competitive performance to ListNet, Algorithm 3 performs quite poorly on Yahoo and Yandex datasets.
on Yahoo dataset (average NDCG@10 over last ten iterations= 0.75 vs 0.74) and has overlapping performance on Yandex dataset. The experiments validate that our proposed perceptron type algorithm (Algorithm 2) has competitive performance to online ListNet on real ranking datasets, even though it does not achieve the theoretical lower bound. Algorithm 3 performs quite poorly on both Yandex and Yahoo datasets, and actually does substantially worse than Algorithm 2 on all three datasets.

9. Conclusion

We proposed two perceptron type algorithms for learning to rank, as analogues of the perceptron for classification algorithm. We showed how, under assumption of separability (i.e., existence of perfect ranker), the cumulative NDCG/AP induced loss is bounded by a constant. The first algorithm operates on a listwise, large margin family of surrogates, which are adaptable to NDCG and AP. The second algorithm is based on another large margin surrogate, which does not have the listwise property. We also proved a lower bound and showed that it is the minimax bound, since the second algorithm achieves the bound. We conducted experiments on simulated and commercial datasets to corroborate our theoretical results.

An important aspect of perceptron type algorithms is that the ranking function is updated only on a mistake round. Since non-linear ranking functions generally have better performance than linear ranking functions, an online algorithm learning a flexible non-linear kernel ranking function would be very useful in practice. We highlight how perceptron’s “update only on mistake round” aspect can prove to be powerful when learning a non-linear kernel ranking function.

Since the score of each document is obtained via inner product of ranking parameter $w$ and feature representation of document $x$, this can be easily kernelized to learn a range of non-linear ranking functions. However, the inherent difficulty of applying OGD to a convex ranking surrogate with kernel function is that at each update step, the document list ($X$ matrix) will need to be stored in memory. For moderately large dataset, this soon becomes a practical impossibility. One way of bypassing the problem is to approximately represent the kernel function via an explicit feature projection [Rahimi and Recht, 2007; Le et al., 2013]. However, even for moderate length features (like 136 for MSWEB10K), the projection dimension becomes too high for efficient computation. Another technique is to have a finite budget $B$ for storing document matrices and discard carefully chosen members from the budget when budget capacity is exceeded. This budget extension has been studied for perceptron in classification [Dekel et al., 2008; Cavallanti et al., 2007]. The fact that perceptron updates are only on mistake rounds leads to strong theoretical bounds on target loss. For OGD on general convex surrogates, the fact that function update happens on every round leads to inherent difficulties when using their kernelized versions [Zhao et al., 2012] (the theoretical guarantees on the target loss are not as strong as in the kernelized perceptron on a budget case). The results presented in this paper open up a fruitful direction for further research: namely, to extend the perceptron algorithm to non-linear ranking functions by using kernels and establishing theoretical performance bounds in the presence of a memory budget.

Acknowledgments
We gratefully acknowledge the support of NSF under grant IIS-1319810. We also thank Prateek Jain for pointing out the relevant question on perceptron bound for NDCG cut-off at \( k \ll m \).

**References**

R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*, volume 463. ACM press New York., 1999.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of International Conference on Machine Learning*, pages 129–136, 2007.

Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

Soumen Chakrabarti, Rajiv Khanna, Uma Sawant, and Chiru Bhattacharyya. Structured learning for non-smooth ranking losses. In *Proceedings of Conference on Knowledge Discovery and Data Mining*, pages 88–96, 2008.

Olivier Chapelle and Yi Chang. Yahoo! learning to rank challenge overview. In *Yahoo! Learning to Rank Challenge*, pages 1–24, 2011.

Olivier Chapelle, Quoc Le, and Alex Smola. Large margin optimization of ranking measures. In *Neural Information Processing Systems Workshop: Machine Learning for Web Search*, 2007.

Koby Crammer and Yoram Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems*, pages 641–647, 2001.

Koby Crammer and Yoram Singer. A new family of online algorithms for category ranking. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 151–158. ACM, 2002.

Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a budget. *SIAM Journal on Computing*, 37(5):1342–1372, 2008.

Jonathan L Elsas, Vitor R Carvalho, and Jaime G Carbonell. Fast learning of document ranking functions with the committee perceptron. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 55–64. ACM, 2008.

Yoav Freund and Robert E Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, pages 277–296, 1999.

Edward F Harrington. Online ranking/collaborative filtering using the perceptron algorithm. In *Proceedings of International Conference on Machine Learning*, volume 20, pages 250–257, 2003.

Prateek Jain, Nagarajan Natarajan, and Ambuj Tewari. A family of online algorithms for general prediction problems. In *Advances in Neural Information Processing Systems*, 2015.

K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, pages 422–446, 2002.
Quoc Le, Tamás Sarlós, and Alex Smola. Fastfoodapproximating kernel expansions in loglinear time. In *Proceedings of International Conference on Machine Learning*, 2013.

T.Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR workshop*, pages 3–10, 2007.

Weijian Ni and Yalou Huang. Online ranking algorithm based on perceptron with margins. In *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, pages 814–819. IEEE, 2008.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, pages 107–194, 2011.

Libin Shen and Aravind K Joshi. Ranking and reranking with perceptron. *Machine Learning*, 60 (1-3):73–96, 2005.

Niek Tax, Sander Bockting, and Djoerd Hiemstra. A cross-benchmark comparison of 87 learning to rank methods. *Information Processing and Management*, pages 757–772, 2015.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, second edition, 1999.

Jialei Wang, Ji Wan, Yongdong Zhang, and Steven CH Hoi. Solar: Scalable online learning algorithms for ranking. In *Proceedings of Association for Computational Linguistics*, 2015.

Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. A support vector method for optimizing average precision. In *Proceedings of ACM SIGIR*, pages 271–278, 2007.

Peilin Zhao, Jialei Wang, Pengcheng Wu, Rong Jin, and Steven CH Hoi. Fast bounded online gradient descent algorithms for scalable kernel-based online learning. *arXiv preprint arXiv:1206.4633*, 2012.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of International Conference on Machine Learning*, pages 928–936, 2003.
Appendix A.

Proof of Theorem

Proof for AP: As stated previously, documents pertaining to every query is sorted according to relevance labels. We point out another critical property of AP (for that matter any ranking measure). AP is only affected when scores of 2 documents, which have different relevance levels, are not consistent with the relevance levels, as long as ranking is obtained by sorting scores in descending order. That is, if \( R_i = R_j \), then it does not matter whether \( s_i > s_j \) or \( s_i < s_j \). So, without loss of generality, we can always assume that within same relevance class, the documents are sorted according to scores. That is, if \( R_i = R_j \) with \( i < j \), then \( s_i \geq s_j \). The without loss of generality holds because SLAM is calculated with knowledge of relevance and score vector. Thus, within same relevance class, we can sort the documents according to their scores (effectively exchanging document identities), without affecting SLAM loss.

Let \( R \in \mathbb{R}^m \) be an arbitrary binary relevance vector, with \( r \) relevant documents and \( m - r \) irrelevant documents in a list. AP loss is only incurred if at least 1 irrelevant document is placed above at least 1 relevant document. With reference to \( \phi^v_{SLAM} \) in Eq. (9), for any \( i \geq r + 1 \) and \( \forall j > i \), we have \( 1(R_i > R_j) = 0 \), since \( R_i = R_j = 0 \). For any \( i \geq r + 1 \) and \( \forall j < i \), \( 1(R_i > R_j) = 0 \) since documents are sorted according to relevance labels and \( R_i = 0, R_j = 1 \). Thus, w.l.o.g., we can take \( v_{r+1}, \ldots, v_m = 0 \), since indicator in SLAM loss will never turn on for \( i \geq r + 1 \).

Let a score vector \( s \) be such that an irrelevant document \( j \) has the highest score among \( m \) documents. Then, \( \phi^v_{SLAM} = v_1(1 + s_j - s_1) + v_2(1 + s_j - s_2) + \ldots + v_r(1 + s_j - s_r) \). The maximum possible AP induced loss in case at least one irrelevant document has higher score than all relevant documents is when all irrelevant documents outscore all relevant documents. The AP loss in that case is: \( 1 - \frac{1}{r} \left( \frac{1}{m-r+1} + \frac{2}{m-r+2} + \ldots + \frac{r}{m-r+r} \right) \). Since \( \phi^v_{SLAM} \) has to upper bound AP \( \forall s \) (for each \( R \)) and since \( s_j \) can be infinitesimally greater than all other score components (thus, \( 1 + s_j - s_i \sim 1, \forall i = 1, \ldots, r \)), we need the following equation for upper bound property to hold:

\[
v_1 + v_2 + \ldots + v_r \geq 1 - \frac{1}{r} \left( \frac{1}{m-r+1} + \frac{2}{m-r+2} + \ldots + \frac{r}{m-r+r} \right).
\]

Similarly, let a score vector \( s \) be such that an irrelevant document \( j \) has higher score than all but the 1st relevant document. Then \( \phi^v_{SLAM} = v_2(1 + s_j - s_2) + v_3(1 + s_j - s_3) + \ldots + v_r(1 + s_j - s_r) \). The maximum possible AP induced loss in case at least one irrelevant document has higher score than all but 1st relevant document is when all irrelevant documents are placed above all relevant documents but first one. The AP loss in that case is: \( 1 - \frac{1}{r} \left( 1 + \frac{2}{m-r+2} + \frac{3}{m-r+3} + \ldots + \frac{r}{m-r+r} \right) \).

Following same line of logic for upper bounding as before, we get

\[
v_2 + v_3 + \ldots + v_r \geq 1 - \frac{1}{r} \left( 1 + \frac{2}{m-r+2} + \frac{3}{m-r+3} + \ldots + \frac{r}{m-r+r} \right).
\]

Likewise, if we keep repeating the logic, we get sequence of inequalities, with the last inequality being

\[
v_r \geq 1 - \frac{1}{r} \left( r - 1 + \frac{r}{m-r+r} \right).
\]

Now, it can be easily seen that our definition of \( v_{AP} \) satisfies the inequalities.

Proof for NDCG:

We once again remind that \( \pi^{-1}(i) \) means position of document \( i \) in permutation \( \pi \). Thus, if document \( i \) is placed at position \( j \) in \( \pi \), then \( \pi^{-1}(i) = j \). Moreover, like AP, we assume that \( R_1 \geq R_2 \geq \ldots \geq R_m \) and that within same relevance class, documents are sorted according to
score. We have a modified definition of NDCG, for \( k = m \), which is required for the proof:

\[
\text{NDCG}(s, R) = \frac{1}{Z(R)} \sum_{i=1}^{m} G(R_i) D(\pi^{-1}_s(i))
\]  

(31)

where \( G(r) = 2^r - 1 \), \( D(i) = \frac{1}{\log_2(i+1)} \), \( Z(R) = \max_{\pi} \sum_{i=1}^{m} G(R_i) D(\pi^{-1}(i)) \). We begin the proof:

\[
1 - \text{NDCG}(s, R) \\
= \frac{1}{Z(R)} \sum_{i=1}^{m} G(R_i) D(i) - \frac{1}{Z(R)} \sum_{i=1}^{m} G(R_i) D(\pi^{-1}_s(i)) \\
= \frac{1}{Z(R)} \sum_{i=1}^{m} G(R_i) (D(i) - D(\pi^{-1}_s(i)))
\]

Now, \( D(i) = \frac{1}{\log_2(i+1)} \) is a decreasing function of \( i \). \( D(i) - D(\pi^{-1}_s(i)) \) is positive only if \( i < \pi^{-1}_s(i) \). This means that document \( i \) in the original list, is placed at position \( \pi^{-1}_s(i) \), which comes after, by order of score vector \( s \). By the assumption that indices of documents within same relevance class are sorted according to their scores, this means that document \( i \) is outscored by another document (say with index \( k \)) with lower relevance level. At that point, the function \( \max(0, \max_{j=1,...,m} \{ 1(R_i > R_j)(1 + s_j - s_i) \}) \) turns on with value at least 1 (i.e., \( 1 + s_k - s_i > 1 \)) and with weight vector \( v^NDCG_i = \frac{G(R_i) D(i)}{Z(R)} \). We can now easily see the upper bound property.