RankDistil: Knowledge Distillation for Ranking

Sashank J. Reddi, Rama Kumar Pasumarthi, Aditya Krishna Menon, Ankit Singh Rawat
Felix Yu, Seungyeon Kim, Andreas Veit, Sanjiv Kumar
Google Research

Abstract

Knowledge distillation is an approach to improve the performance of a student model by using the knowledge of a complex teacher. Despite its success in several deep learning applications, the study of distillation is mostly confined to classification settings. In particular, the use of distillation in top-k ranking settings, where the goal is to rank k most relevant items correctly, remains largely unexplored. In this paper, we study such ranking problems through the lens of distillation. We present a distillation framework for top-k ranking and draw connections with the existing ranking methods. The core idea of this framework is to preserve the ranking at the top by matching the order of items of student and teacher, while penalizing large scores for items ranked low by the teacher. Building on this, we develop a novel distillation approach, RankDistil, specifically catered towards ranking problems with a large number of items to rank, and establish statistical basis for the method. Finally, we conduct experiments which demonstrate that RankDistil yields benefits over commonly used baselines for ranking problems.

1 Introduction

Distillation is the process of using a *teacher* model to improve the performance of a *student* model (Craven and Shavlik, 1995; Breiman and Shang, 1996; Bucilă et al., 2006; Xue et al., 2013; Ba and Caruana, 2014; Hinton et al., 2015). The idea of distillation was originally devised as a way to transfer the knowledge of a complex teacher to a more compact student model, thereby, providing model compression (Bucilă et al., 2006; Hinton et al., 2015). Beyond this model compression view, recent works have also shown its benefit as a training technique wherein a similar-or-larger capacity model is used as a student to improve performance (Furlanello et al., 2018; Xie et al., 2019). Distillation is typically used for classification settings where rather than fitting to raw class labels, the student fits to the "smoothed" *pseudo-labels* obtained from a teacher. In deep learning applications, this amounts to minimizing the softmax cross-entropy loss between the teacher and student logits.

Despite its widespread empirical successes in classification settings, distillation methods for *ranking* problems have been largely unexplored. Unlike classification, ranking problems typically involve learning to rank a list of items for a given context and are, hence, more challenging. Such a setting is often referred to as "learning to rank" and has been extensively studied in information retrieval and machine learning communities (Joachims, 2002; Cao et al., 2007; Xia et al., 2008). A prototypical example of this setting is document ranking where the context is a query and the items are documents (Joachims, 2002). This is achieved by learning a ranking function that maps a query to a list of the documents sorted by relevance to the query. Given the prevalence of ranking problems in machine learning, a principled study of distillation approaches for this setting is important.

To this end, in this paper, we study distillation approaches for ranking problems. The ranking setting of our interest has two key aspects: (1) the number of items to rank, K, is large and (2) ranking at the first k positions is crucial. Typically, $k \ll K$ for many ranking problems that arise in machine learning. The presence of a large number of items makes distillation for this setting particularly challenging. Furthermore, it is unclear what constitutes a ranking equivalent of pseudo-labels used for distillation in classification settings. To tackle these issues, we develop a novel approach for distillation for ranking problems. In summary, our main

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

contributions are as follows:

(i) We study a distillation framework for top-k ranking problems. The key idea of this framework is to preserve the ranking at the top by matching the order of items of student and teacher, while penalizing items ranked low by the teacher. Using this framework, we draw connections with existing algorithms and present them in a unified view.

(ii) Building on this, we develop, RankDistil, a novel algorithm for distillation for ranking problems. By borrowing ideas from negative sampling/mining literature in classification, we show RankDistil scales seamlessly to ranking settings with a large number of items to rank. Furthermore, we also establish statistical basis for RankDistil.

(iii) We experimentally validate the value of our approach for both the tasks of ranking distillation and as a ranking loss. We compare with several baselines typically used in "learning to rank" setting. Our results show that RankDistil outperforms these baselines on popular ranking metrics such as NDCG and MRR.

Related Work. The literature on distillation and ranking problems is vast and hence, we only discuss the works that are most relevant to our paper.

Knowledge Distillation. Initial works on distillation focused on model compression. In a seminal work, Bucilă et al. (2006) demonstrated an approach to learn a single neural network to mimic an ensemble of neural networks. Subsequently, Ba and Caruana (2014) showed increase in learning performance of shallow neural networks, by training them to mimic deep neural networks. There has been increased interest in distillation following the work of (Hinton et al., 2015), who proposed minimizing the softmax cross entropy between student and teacher logits. Thereafter, several works later studied the utility of distillation in various settings (Furlanello et al., 2016; Czarnecki et al., 2017; Lopez-Paz et al., 2016; Menon et al., 2020). More recent works have demonstrated the benefit of distillation as a training technique (Furlanello et al., 2018; Xie et al., 2019). However, most of these works in supervised learning focus only on classification, which is easier than the ranking setup of our interest.

Ranking problems. Earlier works on ranking problems focused on pairwise loss functions (Herbrich et al., 1999; Burges et al., 2005; Crammer and Singer, 2001). Pairwise approaches to ranking use classification of pair of items into two classes – correctly and incorrectly ranked (e.g. ranking SVM (Herbrich et al., 1999; Joachims, 2002)). Generalization of pairwise approach based on list of items later became popular in information retrieval and machine learning (Cao et al., 2007; Xia et al., 2008). These approaches are usually based on probabilistic models for ranking used in statistics (see e.g. seminal work of Luce (Luce, 1959) and Plackett (Plackett, 1975)). However, these approaches are typically unsuitable for ranking of large set of items of our interest (Luo et al., 2015). Furthermore, these methods are typically not geared towards distillation settings. To our knowledge, Tang and Wang (2018); Gao et al. (2020) are the only works that study distillation for general ranking problems. However, both these work study only a specific position-aware binary loss for distillation. Furthermore, Tang and Wang (2018) discard items ranked low by the teacher, which typically degrades performance. Building upon (Tang and Wang, 2018), Lee et al. (2019) propose a distillation approach for collaborative filtering, which is different from the setting we consider in the paper. In fact, as we shall see later, these works are simple instances of our framework (see Section 3.4). Thus, these works focus on a specific loss function or discard valuable information, which are addressed by our work in a principled manner.

2 General Framework for TOP-k Ranking

Notation. We use v_i to denote the i^{th} element of any vector v. For a vector $v \in \mathbb{R}^K$ and subset $S \subseteq [K]$, v_S denotes vector $(v_i)_{i \in S}$. We use $\text{Top}_k(v)$ to denote the indices of largest k elements of vector v (with ties broken in ascending order of the indices). For any index set $S \subseteq [K]$, we use $\text{Top}_{k,S}(v)$ to denote the largest k indices of vector v that are in S. For $v \in \mathbb{R}^K$, $\text{sort}(v)$ is used to denote the indices corresponding to values of v sorted in *descending* order. Also, $\text{argmax}_i v$ is used to denote the index of i^{th} largest element of v (again breaking ties in ascending order of the indices). For a vector v, we use $v_{[i]}$ to denote the i^{th} largest element of v. For a distribution Q, $supp(Q)$ represents its support. Finally, for sets S_1 and S_2, we use $S_1 - S_2$ to denote the set difference of S_1 and S_2. For a set $S \subseteq [K]$, $\mathcal{P}(S)$ denotes set of all permutations of S.

Problem Setting. In the classical ranking setting, we are given a training sample $\mathcal{S} \doteq \{(x_i, y_i)\}_{i=1}^n \sim \mathbb{P}^n$, for unknown distribution $\mathbb{P}$ over instances $\mathcal{X}$ and relevance labels $\mathcal{Y} \subseteq \mathbb{R}^K$ (Tewari and Chaudhuri, 2016). Note that the sorted indices of values in label y naturally define a ranking in $\mathcal{P}(K)$ where $\mathcal{P}(K)$ is the set of all permutations of $\{1, 2, \ldots, K\} \triangleq [K]$; however, the relevance scores and the corresponding ranking are typically noisy. This setting contrasts with the standard classification setting where the labels are categorical. Our goal is to learn a predictor $f \colon \mathcal{X} \to \mathbb{R}^K$ so as to

minimize the *risk* of f,

$$R(f) \doteq \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{P}} [\phi_{0-1}(x, y, f(x))] \,. \tag{1}$$

Here, ϕ_{0-1} is the following loss function:

$$\phi_{0-1}(x, y, f(x)) = \begin{cases} 0, & \text{TOP}_k(y) = \text{TOP}_k(f(x)) \\ 1, & \text{otherwise.} \end{cases}$$

We use f^* to denote a minimizer of the function $R(f)$. Since this loss function is discontinuous and non-convex, surrogate loss functions that are easy to optimize are used (Xia et al., 2009). In this paper, we use ϕ to denote a surrogate loss function. In particular, $\phi\colon \mathcal{X} \times \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}_+$ is a loss function, where for label $y \in \mathbb{R}^K$ and prediction vector $f(x) \in \mathbb{R}^K$, $\phi(x, y, f(x))$ is the loss incurred for predicting $f(x)$ when the true label is y. For the ease of exposition, we assume the following about f^*.

Assumption 1. The top $(p+1)$ (where $p \geq k$) values of $f^*(x)$ are unique for all $x \in \mathcal{X}$ i.e., $[f^*(x)]_i \neq [f^*(x)]_j$ for all $i \neq j$, $i, j \in \text{TOP}_{p+1}(f^*(x))$ and $x \in \mathcal{X}$.

This assumption is mainly for simplicity in stating the theoretical results in the presence of ties and does not alter the conceptual nature of our results. Our ranking setting differs from the classical ranking setting in one key manner: we are interested in partial rankings and in particular, ranking at the first k positions (see e.g. (Xia et al., 2009)). Thus, we measure the success in terms of ranking metrics at the top k positions. Also, recall that K is assumed to be large. Both of these aspects are fairly common in machine learning and information retrieval settings (Joachims, 2002; Luo et al., 2015). In the distillation setting of our interest, we have access to a *teacher model*, f^{t} which is usually a more complex model or an ensemble of models. We assume that f^{t} is bounded from below. This model is assumed to have good performance in terms of ranking measures like MRR, NDCG (see (Chen et al., 2009) for more details on these measures) and our goal is to mimic the teacher in the sense of ranking of the top k positions. To formalize this, we define the following.

Definition 1. *A prediction function $f : \mathcal{X} \to \mathcal{Y}$ is said to be k-compatible if $\text{argmax}_i f(x) = \text{argmax}_i f^*(x)$ for all $i \in [k]$ and $x \in \mathcal{X}$. A loss function $\phi : \mathcal{X} \times \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}_+$ is said to be k-consistent if $\text{argmax}_i f_\phi(x) = \text{argmax}_i f^*(x)$ where $f_\phi(x) \in \text{argmin}_{s\in\mathbb{R}^K} \mathbb{E}_{y|x} \phi(x, y, s)$ for all $x \in \mathcal{X}$.*

Recall that $\text{argmax}_i v$ denotes the index of i^{th} largest element of vector v and f^* is the minimizer of $R(f)$ (Equation 1). We observe that while k-compatibility is a property of a *prediction* function, k-consistency is a property of a *loss* function. This definition essentially states that a loss function is consistent if for any $x \in \mathcal{X}$, the prediction function that minimizes that loss function is k-compatible. Note that the definition is based on the first k positions and thus, the ranking of items after the k positions does not affect our consistency results. With slight abuse of terminology, for any $x \in \mathcal{X}$, we refer to the indices corresponding to top p scores of the teacher, $\text{TOP}_p(f^{\text{t}}(x))$ (where $p \geq k$), as "positives" in our paper. The rest of the indices are referred to as "negatives". For any x, a natural segregation of items into two buckets — top-p items and the rest — makes this notation appropriate for our setting.

2.1 Distillation loss for TOP-k Ranking

In this section, we develop a general framework for distillation of ranking problems. In particular, we define a family of loss function that are well-suited for the setting of our interest.

Definition 2. *Let $\ell_d : \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}_+$ be a loss function, $f_d(x) \in \text{argmin}_{s\in\mathbb{R}^K} \ell_d(f^{\text{t}}(x), s)$. The loss function ℓ_d is called p-focused distillation loss (p-FDL) for f^{t} if the following conditions hold:*

1. *The top p items according to $f_d(x)$ and $f^{\text{t}}(x)$ and their order match i.e., $\text{argmax}_i f_d(x) = \text{argmax}_i f^{\text{t}}(x)$ for all $i \in [k]$ and $x \in \mathcal{X}$.*
2. *The score $[f_d(x)]_{[p+1]} < [f_d(x)]_{[p]}$ for all $x \in \mathcal{X}$.*

The first condition states that the top p items according to the teacher scores and the minimizer f_d exactly match. The second condition, when combined with the first, ensures that the scores of the items that are ranked low by the teacher are penalized in f_d. Note that for FDL, the order of the items not at the top is not important; thus, the order of items not in the top may necessarily be preserved. When ranking performance of the teacher is good, by satisfying these conditions, we can ensure good student performance with respect to top-k ranking measures. The simple result described below formalizes this intuition. Note that the distillation loss ℓ_d that satisfies the above conditions only depends on the teacher score and student score. Thus, we are not using the labels from our training dataset $\mathcal{S}$. However, we can easily modify our distillation procedure to use the training data labels in addition to the teacher scores. The following is a straightforward observation regarding FDL loss functions.

Claim 3. *Suppose Assumption 1 holds and the teacher predictor f^{t} is k-compatible. Then, loss function $\phi_d(x, y, s) = \ell_d(f^{\text{t}}(x), s)$, where ℓ_d is a p-FDL loss, is k-consistent when $p \geq k$.*

This claim establishes the statistical consistency of the FDL family of loss functions when the teacher

Algorithm 1 RankDistil

Initialization: Initial predictor f, Teacher predictor f^{t}, negative sampling distribution $Q(.|t)$, loss function $\ell_{\text{RankDistil}}$, integer $p \geq k$, batch size m, mined batch size $b \leq m$
for $h = 0, \cdots, R-1$ **do**
  Uniformly randomly select an example $\{x, y\}$
  Sample index set B of size m using the distribution $Q(.|f^{\mathrm{t}}(x))$
  Compute $P = \text{Top}_p(f^{\mathrm{t}}(x))$ and $N = \text{Top}_{b,B}(f(x))$
  Compute $s_l = f_l(x)$ for $l \in P \cup N$
  Compute $g_h = \nabla \ell_{\text{RankDistil}}(f^{\mathrm{t}}(x), s, P, N)$ (e.g. Equation 3 & 4)
  Update predictor f using the gradient g_h
end for

predictor is compatible (as per Definition 1). While FDL provides a basic framework to study distillation losses for ranking problems, it is not clear what loss functions satisfy these conditions and, more generally, how such loss functions can be optimized efficiently when K is large. Next, we delve into these questions in more detail and provide concrete instantiations of our framework.

3 RankDistil

Building upon our framework in the previous section, we propose a novel distillation procedure, RankDistil, for ranking settings where the number of items K is large. To describe the RankDistil algorithm, we define the loss function $\ell_{\text{RankDistil}}(t, s, P, N)$, where $s, t \in \mathbb{R}^K$ and $P, N \subseteq [K]$. Here P and N represent a set of indices which will be used as "positives" and "negatives" respectively in the loss function. The vectors t and s are used to denote teacher's and student's output vectors for a particular training sample respectively. As we shall see in specific instantiations, the loss function $\ell_{\text{RankDistil}}(t, s, P, N)$ aims to

- match the order of scores s_P and t_P i.e., preserve the score order (or equivalently rank) for items in the positive set P, and
- penalize large student scores s_N, but the order of items within the negative set N is not necessarily preserved.

The overall distillation loss (RankDistil-Loss) function $\ell_d(t, s)$ is defined as

$$\mathbb{E}_{B \sim Q^m(.|t)} \left[\ell_{\text{RankDistil}}(t, s, \text{Top}_p(t), \text{Top}_{b,B}(s)) \right],$$

where $B \subseteq [K]$ such that $|B| = m$, $b \leq m$ and $Q(.|t)$ is a categorical distribution over $[K]$ with support $[K] - \text{Top}_p(t)$. Recall that $\text{Top}_{b,B}(s)$ represents the top b indices of score vector s that are in B. RankDistil-Loss is the population expectation of $\ell_{\text{RankDistil}}$ where P and N are chosen to be $\text{Top}_p(t)$ and a subset of m random samples drawn from Q respectively. Algorithm 1 provides the pseudo-code for optimizing this loss function. The key idea of the algorithm is to get an unbiased estimate of gradient of RankDistil-Loss. This is accomplished by randomly sampling m negatives according to the distribution Q and picking the items with largest b scores (where $b \leq m$) according to the student score s to obtain the negative set used in RankDistil-Loss. The distribution Q is used to sample the negative set since K is large. The strategy of picking the b largest scoring negatives is similar to the one explored by Reddi et al. (2019) in the classification setting.

Overall, the time complexity of each iteration of the algorithm is $O((p+m)C(p,m))$ (where $C(p,m)$ is the time for computing gradient of $\ell_{\text{RankDistil}}$) and is, thus, independent of K. Since we can compute an unbiased estimate of the gradient in a cheap manner, RankDistil-Loss can be optimized efficiently. Note that computing top-p items of $f^{\mathrm{t}}(x)$ in Algorithm 1 takes $O(K)$; however, since it is computed only once and can be done offline, we ignore it in time complexity analysis.

Before we look at different settings for $\ell_{\text{RankDistil}}$, it is instructive to examine a simple concrete example of RankDistil. Consider the following modified variant of softmax cross-entropy:

$$\ell_{\text{RankDistil}}(t, s, P, N) = -\sum_{j \in P} \frac{\exp(t_j)}{\sum_{l \in P} \exp(t_l)} \log \left(\frac{\exp(s_j)}{\sum_{l \in P \cup N} \exp(s_l)} \right). \quad (2)$$

Note that the loss function only depends on the scores s_P, t_P and s_N and is thus, efficient to compute when P and N are small. The set N is sampled from a distribution $Q(.|t)$. A simple instance of Q is a uniform categorical distribution whose support is $[K] - \text{Top}_p(t)$. However, more complex distributions can be used to improve the optimization and generalization performance. In the subsequent sections, we consider three families of RankDistil-Loss which differ in the way scores s_P and s_N interact in the loss function.

3.1 Coupled RankDistil-Loss

We now consider the case where the scores s_P and s_N in $\ell_{\text{RankDistil}}$ are coupled, i.e., the loss function is *not* separable with respect to these scores. We define the following probability model.

Definition 4 (r-Plackett's Probability Model)**.** *Suppose $S \subseteq [K]$ such that $|S| \geq r$. Then, for $\pi \in \mathcal{P}(S)$, the probability of the permutation π given the scores $s \in \mathbb{R}^K$ is given by*

$$\mathbb{P}_s(\pi|S) = \frac{1}{(|S|-r)!} \prod_{j=1}^{r} \frac{\exp(s_{\pi(j)})}{\sum_{l=j}^{|S|} \exp(s_{\pi(l)})}.$$

The r-Plackett probability model generalizes the Plackett probability model for ranking (Luce, 1959; Plackett, 1975). The following result shows the r-Plackett model indeed defines a probability distribution over permutations over subsets of $[K]$.

Proposition 5. *Suppose $\mathbb{P}_s$ satisfies r-Plackett's Probability Model, then $\sum_{\pi \in \mathcal{P}(S)} \mathbb{P}_s(\pi|S) = 1$ for all $S \subseteq [K]$ and $|S| \geq r$.*

Using this probability model, we define the coupled loss function as:

$$\begin{aligned}
&\ell_{\text{RankDistil}}(t, s, P, N) \\
&= - \sum_{\pi \in \mathcal{P}(P \cup N)} \mathbb{P}_{\tau(\alpha t, P)}(\pi|P \cup N) \log \mathbb{P}_s(\pi|P \cup N) \\
&= \mathbb{E}_{\pi \sim \mathbb{P}_{\tau(\alpha t, P)}(.|P \cup N)} \left[-\log \mathbb{P}_s(\pi|P \cup N) \right]. \qquad (3)
\end{aligned}$$

where $\alpha > 0$ is the inverse temperature of softmax probability and τ is a threshold function such that, for any $s \in \mathbb{R}^K$ and $S \subseteq [K]$, $\tau_i(s, S) = s_i$ if $s \in S$ and $-M$ otherwise (where M is a large positive real number) or identity function i.e., $\tau_i(s, S) = s_i$ for all $i \in [K]$. The loss is essentially cross-entropy between r-Plackett's probability of student and (possibly thresholded) teacher.

Theorem 6. *Suppose Assumption 1 holds, $r \geq 1$, $p \geq k$, $supp(Q(.|s)) = [K] - \text{Top}_p(s)$ and τ is thresholding function with sufficiently large M. Then* RankDistil-Loss *with $\ell_{\text{RankDistil}}$ as defined in Equation 3 is p-FDL.*

The result above shows consistency of our coupled loss function. Note that $\ell_{\text{RankDistil}}$ defined in Equation 3 requires iterating over all permutations $\mathcal{P}(P \cup N)$. However, this can be computed in $O(p^r r + m)$ when τ is thresholding function with $M = \infty$ (see Appendix A.2 for more details). When $r = 1$ and p is small, this computation is tractable. In the general case, we resort to a Monte Carlo approximation by sampling permutations from the corresponding r-Plackett's model probability model (Definition 4). For the purpose of clarity, the complete algorithm is provided in Appendix A.

Coupled RankDistil-Loss proposed here has connections with many existing algorithms. When $r = 1$, this reduces to the modified variant of softmax cross-entropy defined in Equation 2. Note that this also corresponds to a distillation variant of ListNet (Cao et al., 2007) with the distinction about handling of negatives set N and that the scores are only matched for the top p positions. Our coupled loss function also captures other generalizations of ListNet (e.g. see (Xia et al., 2009; Luo et al., 2015)) as special cases. Furthermore, $p = K$ and $\alpha \to \infty$ corresponds to our distillation variant of ListMLE loss function (Xia et al., 2008). Note that high α corresponds to low temperature regime of softmax probability. Thus, coupled RankDistil-Loss generalizes several existing loss function.

3.2 Binary RankDistil-Loss

Next, we consider a separable case where P and N can be decomposed.

$$\ell_{\text{RankDistil}}(t, s, P, N) = \Psi(t, s, P) + \sum_{i \in N} \varphi(-s_i), \quad (4)$$

where Ψ is a (multiclass-like) loss function such that $s_P = t_P$ at the minimizer and φ is any binary classification loss function. As seen in Equation 4, the loss function naturally decomposes over the sets N and P. This family captures a wide range of loss functions. Table 1 shows a list of possible options for Ψ and φ. This table covers a few popular options and is not exhaustive. The following simple result can be shown for binary RankDistil-Loss.

Theorem 7. *Suppose Assumption 1 holds, $p \geq k$ and $supp(Q(.|s)) = [K] - \text{Top}_p(s)$. Also, suppose $[f^t(x)]_{[p+1]} < \gamma$ for all $x \in \mathcal{X}$. Then* RankDistil-Loss *with $\ell_{\text{RankDistil}}$ in Eq 4 is p-FDL when*

1. *We have $\text{argmax}_i s^*_P = \text{argmax}_i f^t_P(x)$ for all $x \in \mathcal{X}$, $P \subseteq [K]$, $i \in [p]$ where $s^* \in \text{argmin}_s \Psi(f^t(x), s, P)$ and,*
2. *The function $\varphi : \mathbb{R} \to \mathbb{R}_+$ is a non-increasing and strictly decreasing on $(-\infty, -\gamma]$.*

This result essentially shows consistency of all the loss functions listed in Table 1. The case where Ψ is q-Regression (with $q = 2$) and φ is hinge loss is of particular interest. This is the ranking equivalent of logit matching used in usual classification setting (Ba and Caruana, 2014; Hinton et al., 2015). Also, when $p = 1$, Ψ is 2-Regression, φ is square hinge loss and scores are in $[-1, 1]$, the loss corresponds to a distillation variant of the cosine contrastive loss used in classification.

3.3 Pairwise RankDistil-Loss

In this section, we consider pairwise RankDistil-Loss functions. For this family of loss functions, $\ell_{\text{RankDistil}}$

Table 1: Example instantiations of Binary RankDistil-Loss. Here q is a positive real number in q-Regression and γ is the margin in Hinge loss. These instantiations of Ψ and φ can also be used for the pairwise RankDistil-Loss (see Equation 5).

$\Psi(t, s, P)$		$\varphi(s_i)$	
Softmax CE	$\sum_{i\in P}\left\{-\frac{e^{t_i}}{\sum_{j\in P} e^{t_j}} \cdot \log \frac{e^{s_i}}{\sum_{j\in P} e^{s_j}}\right\}$	Logistic	$\log(1+e^{-s_i})$
Sigmoid CE	$\sum_{i\in P}\sum_{z\in\{-1,1\}} -\frac{1}{1+e^{zt_i}} \cdot \log \frac{1}{1+e^{zs_i}}$	Hinge	$\max\{0, \gamma - s_i\}$
q-Regression	$\|t_P - s_P\|_q^q$	Square Hinge	$\max\{0, \gamma - s_i\}^2$

is of the following form:

$$\ell_{\text{RankDistil}}(t, s, P, N) = \Psi(t, s, P) + \sum_{i\in N}\sum_{j\in P} \varphi(s_j - s_i). \tag{5}$$

In contrast to the binary RankDistil-Loss, these loss functions use pairwise comparisons between the items in P and N. We can essentially use the same Ψ and φ as listed in Table 1. The following result shows that this class of loss functions are p-FDL under fairly mild conditions.

Theorem 8. *Suppose Assumption 1 holds, $p \geq k$ and $supp(Q(.|s)) = [K] - \text{Top}_p(s)$. Then* RankDistil-Loss *with $\ell_{\text{RankDistil}}$ as defined in Equation 5 is p-FDL when*

1. *We have $\text{argmax}_i s^*_P = \text{argmax}_i f^{\text{t}}_P(x)$ for all $x \in \mathcal{X}$, $P \subseteq [K]$, $i \in [p]$ where $s^* \in \text{argmin}_s \Psi(f^{\text{t}}(x), s, P)$ and,*
2. *The function $\varphi : \mathbb{R} \to \mathbb{R}_+$ is a non-increasing and strictly decreasing on $(-\infty, 0]$.*

For pairwise RankDistil-Loss, in addition to the loss functions in Table 1, pairwise comparison based loss can also be used for $\Psi(t, s, P)$ in Equation 5. In particular, we can use Ψ of the form:

$$\Psi(t, s, P) = \mathbb{1}(t_i - t_j)\varphi(s_i - s_j) + \mathbb{1}(t_j - t_i)\varphi(s_j - s_i) \tag{6}$$

where φ is as defined in Table 1 and $1(x)$ is the indicator function. Here, the exact scores of the teacher are not used, rather just the order of items in P based on teacher's score is used. Also, pairwise RankDistil-Loss decomposes over the P and N and furthermore, can be decomposed over pairs of items. Such a property typically makes it amenable to efficient optimization. Note that the assumption on f^{t} used in Theorem 7 is not required for the pairwise setting.

3.4 Discussion

Relation with previous works. We would like to briefly mention the distinction with existing literature. Tang and Wang (2018) and Ranker Distill in Gao et al. (2020) are the most relevant to our work. At a high level, both these works can be seen as simple instances of binary RankDistil-Loss with Ψ being sigmoid cross-entropy. Note that Tang and Wang (2018) uses a combination of loss on relevance labels and ranking distillation loss as their final loss. While we do not utilize relevance labels during distillation, it is straightforward to incorporate it in our framework. We will use these methods as baselines in our empirical analysis.

When employing distillation in a top-k ranking setting, access to an accurate teacher model provides a clean classification of items as positives and negatives. Such a classification is typically not reliable in settings with *noisy* relevance labels (e.g. (Cao et al., 2007; Xia et al., 2008)). This natural partition of items is exploited in RankDistil to design generic loss functions that are well-aligned with our goal (e.g. refer to the role of sets P and N in RankDistil-Loss). Furthermore, the presence of a large set of items to rank poses a unique challenge in top-k ranking setting. RankDistil presents a novel negative sampling/mining approach to handle ranking setting for large K. To our knowledge, this is the first work to propose a general strategy for distillation in large-scale top-k ranking settings.

Position-aware loss. Finally, note that loss functions in the paper treat the first p positions equally. In practice, using a discount factor to weigh each position differently aligns well with metrics like NDCG. This can be easily incorporated in our framework (e.g. q-Regression can be modified to $\sum_{i=1}^{p} \beta^{i-1}|t_{\pi(i)} - s_{\pi(i)}|^q$ where $\pi = \text{sort}(t)$ and $\beta \in (0, 1]$). All our experiments use such a discount factor.

4 Experiments

In this section, we present empirical results for RankDistil. We focus on two aspects in our empirical analysis: (i) performance gain by using RankDistil for distillation and ranking purpose, and (ii) comparison of RankDistil-Loss proposed in the paper with standard ranking losses.

Table 2: Statistics of ranking datasets. Documents indicates number of documents across all queries and not number of unique documents.

Dataset	Train		Validation		Test	
	# queries	# documents	# queries	# documents	# queries	# documents
MSLR-WEB30K	18,919	2,270,296	6,306	747,218	6,306	753,611
YAHOO LETOR	19,944	473,134	2,994	71,083	6,983	165,660
Istella	20,317	6,410,040	2,902	915,585	9,799	3,129,004
MSMARCO	479,484	430,215,247	53,276	47,801,694	6,971	6,668,940

Table 3: Performance comparison of RankDistil for distillation task on MSLR WEB30K and Yahoo LETOR. The student model is a simple linear model with 128 hidden units and the teacher is a 3-layer FC-BN-ReLU model of hidden units of sizes 1024,512,256. △ indicates statistically significant increase of RankDistil compared to best baseline using paired t-test with significance level 0.05.

Distillation Loss	MSLR WE30K				YAHOO LETOR			
	$NDCG_1$	$NDCG_5$	$NDCG_{10}$	MRR	$NDCG_1$	$NDCG_5$	$NDCG_{10}$	MRR
Teacher	0.4725	0.4668	0.4877	0.8777	0.6921	0.7221	0.7637	0.9433
$\text{Student}_{\text{relevance}}$	0.4248	0.4404	0.4632	0.8654	0.6231	0.6749	0.7261	0.9272
Tang and Wang (2018)	0.4136	0.4276	0.4512	0.8584	0.6474	0.6835	0.7311	0.9204
Gao et al. (2020)	0.3382	0.3654	0.3654	0.8294	0.6269	0.6616	0.7094	0.9113
RankDistil_p	**0.4576**$^{\triangle}$	**0.4573**$^{\triangle}$	**0.4792**$^{\triangle}$	**0.8782**$^{\triangle}$	0.6751	0.7103	0.7503	**0.9445**$^{\triangle}$
RankDistil_c	0.4515	0.4537	0.4757	0.8727	**0.6936**$^{\triangle}$	**0.7205**$^{\triangle}$	**0.7622**$^{\triangle}$	0.9415

RankDistil Loss functions. We compare RankDistil-Loss with other popular ranking losses. To ensure diversity in the loss functions, we use loss functions from coupled, binary and pairwise RankDistil-Loss functions. For the binary case, we pick Ψ and φ as sigmoid cross-entropy and logistic loss respectively. We denote this as RankDistil_b. The pairwise case of RankDistil-Loss (denoted by RankDistil_p) uses logistic loss for both Ψ and φ in Equation 6. Finally, the coupled case (denoted by RankDistil_c) uses the loss function in Equation 3. These loss functions are compared against popular ranking losses used in the literature. In particular, we consider RankNet (Burges et al., 2005), ListNet (Cao et al., 2007) and ListMLE (Xia et al., 2008).

Experimental Setup. We conduct ranking distillation experiments on four datasets: MSLR-WEB30K (Fold1) (Qin and Liu, 2013), Yahoo! Learning to Rank (LETOR) challenge (Set1) (Chapelle and Chang, 2011), and Istella LETOR full dataset (Dato et al., 2016) and MSMARCO passage ranking task (Nguyen et al., 2016). These experiments are designed to look at the effectiveness of RankDistil for ranking distillation task, where a *complex* teacher is used for distillation to learn a simpler model.

For baselines, we compare RankDistil for ranking distillation task with Gao et al. (2020), Tang and Wang (2018). Both these works are specifically catered to the ranking distillation setting of our interest and are, thus, the most relevant baselines. We also compare our performance with teacher model (for headroom analysis) and a student model trained on graded relevance or clicks. Gao et al. (2020) uses a distilled BERT model with a sigmoid cross entropy loss over all of teacher scores. Tang and Wang (2018) combines a sigmoid cross entropy loss over ground-truth relevance scores with a distillation loss of sigmoid cross entropy loss applied only over top-k. As discussed in Section 3.4, both these losses are instances of binary RankDistil-Loss. For the task of ranking loss, we compare RankDistil-Loss with standard ranking losses used in literature such as ListNet, ListMLE, RankNet etc. We would like to emphasize that the model architectures for RankDistil and the corresponding baseline are exactly the same, and they only differ in the nature of the distillation loss for ranking.

The performance is measured in terms of ranking metrics. In particular, we use *Normalized Discounted Cumulative Gain* (NDCG) which is typically used for measuring ranking performance, and used for all comparisons. We describe the statistics of these datasets in Table 2. For MSMARCO dataset, we split the training data into a random $90-10\%$ split for training and validation, respectively. For RankDistil_c, 100 samples from Plackett's model were used for Monte Carlo approximation. A uniform categorical distribution Q

Table 4: Performance comparison of RankDistil for distillation task on MSMARCO passage ranking dataset and Istella dataset. The student model for MSMARCO dataset is Bert-Tiny, which is distilled using ranking losses from a Bert-Base ranking teacher. The student model for Istella dataset is a simple linear model with 128 hidden units and the teacher is a 3-layer FC-BN-ReLU model of hidden units of sizes 1024,512,256. $\triangle$ indicates statistically significant increase of RankDistil compared to best baseline using paired t-test with significance level 0.05.

Distillation Loss	MSMARCO				Istella			
	$NDCG_1$	$NDCG_5$	$NDCG_{10}$	MRR	$NDCG_1$	$NDCG_5$	$NDCG_{10}$	MRR
Teacher	0.7228	0.8423	0.8548	0.8185	0.6458	0.6279	0.6844	0.9536
$\text{Student}_{\text{clicks}}$	0.6031	0.7460	0.7671	0.7221	0.5688	0.5336	0.5779	0.8728
Tang and Wang (2018)	0.6190	0.7643	0.7827	0.7373	0.5483	0.4895	0.5246	0.8306
Gao et al. (2020)	0.6275	0.7684	0.7871	0.7428	0.5491	0.4846	0.5201	0.8181
RankDistil_p	0.6600	0.7974	0.8144	0.7715	0.5066	0.4806	0.5276	0.8257
RankDistil_c	**0.6839**$^\triangle$	**0.8102**$^\triangle$	**0.8264**$^\triangle$	**0.7871**$^\triangle$	**0.5820**$^\triangle$	**0.5395**$^\triangle$	**0.5798**$^\triangle$	**0.8790**$^\triangle$

is used in our experiments for RankDistil. In all experiments, the values of p, r, b and decay factor β were tuned to give the best performance on a held out validation dataset.

MSLR WEB30K, Yahoo LETOR and Istella. All of these datasets comprise of around 30K query-document pairs with 136, 700 and 200 dense features per query-document pair respectively (see Table 2 for more details). Each query-document pair is rated with a graded relevance score from 0 (not relevant) to 4 (highly relevant). For the task of ranking distillation, a 3-hidden layer feedforward network of layer sizes (1024,512,256) with ReLU activation and batch normalization (Ioffe and Szegedy, 2015) is used to learn a teacher model using ListNet loss. A simple linear model with 128 units is used to learn a student model. For the task of ranking loss, we use the 3 layer model used in previous experiment to learn ranking model and report performance of RankDistil-Loss_c, which performs best among variants of RankDistil-Loss. In both experiments, we also apply batch normalization on the inputs, which was found to be effective. The value of m was set to 200.

MSMARCO. The MSMARCO passage ranking dataset contains around 1M anonymized Bing search queries with top 1000 relevant passages retrieved by BM25 score (see Table 2). The objective is to rank the passages by relevance to the query. We use a large Bert (Devlin et al., 2018) based model as a teacher for this setting. In particular, Bert-Base, consisting of a stack of 12-layer 12-head Transformer of hidden size of 768, and a feedforward network of hidden layer size 3072, is used as the teacher model. The transformer block uses *gelu* activations. Query and passages are tokenized using wordpieces to a maximum sequence length of 64, concatenated and passed as inputs to the teacher or student model. ListNet ranking loss was used to train student on clicks and the teacher as it gave the best performance among baseline ranking losses. The teacher is trained for 200K steps using Adagrad (Duchi et al., 2011) with a batch size of 8 and a learning rate of 10^{-5}.

The student model in this setting is Bert-Tiny (Turc et al., 2019), which is similar to the teacher, but of much smaller size. Bert-Tiny consists of 2-layer 2-head Transformer with hidden size of 128, and feedforward network of hidden layer size 512. The student is trained using Adagrad optimizer with learning rate 10^{-4} for 1M steps. The value of m in Algorithm 1 is set to 50. An identity function τ was used in this experiment. The values of p, r and decay factor β were tuned for RankDistil losses. These values were chosen such that they give the best performance on a held out validation set. We use TF-Ranking (Pasumarthi et al., 2019) to implement ranking losses, metrics and training the models.

Ranking Task. Finally, we also compare the effectiveness of RankDistil-Loss as a ranking loss on MSLR-WEB30K and Yahoo! Learning to Rank (LETOR). This experiment is designed to understand the value of RankDistil in settings where graded relevance score is used instead of teacher scores. We do not use MSMARCO dataset for this task since it only consists of binary relevance scores. For this task, we compare RankDistil-Loss against popular ranking losses in the literature. In particular, we use RankNet (Burges et al., 2005), ListNet (Cao et al., 2007) and ListMLE (Xia et al., 2008) for our baseline comparison.

4.1 Model Effectiveness

For ranking distillation task, Tables 3 and 4 provides the comparison of variants of RankDistil with baseline approaches on the test set for LETOR datasets and MSMARCO dataset respectively. We observe that RankDistil significantly outperforms baseline

Table 5: Performance comparison of RankDistil-Loss for ranking task on MSLR WEB30K and Yahoo LETOR. The ranking model is a 3 layer FC-BN-ReLU model of hidden units of sizes 1024,512,256. △ indicates statistically significant increase of RankDistil compared to best baseline using paired t-test with significance level 0.05.

Ranking Loss	MSLR WE30K			YAHOO LETOR		
	$NDCG_1$	$NDCG_5$	$NDCG_{10}$	$NDCG_1$	$NDCG_5$	$NDCG_{10}$
Sigmoid Cross Entropy	0.4765	0.4685	0.4847	0.6513	0.6988	0.7462
RankNet	0.4605	0.4667	0.4898	0.6826	0.7195	0.7623
ListMLE	0.4659	0.4648	0.4853	0.6899	0.7216	0.7628
ListNet	0.4768	0.4679	0.4879	0.6821	0.7125	0.7566
RankDistil-Loss$_c$	**0.4807**$^{\triangle}$	**0.4771**$^{\triangle}$	**0.4953**$^{\triangle}$	**0.6977**$^{\triangle}$	**0.7267**$^{\triangle}$	**0.7669**$^{\triangle}$

Table 6: Number of parameters of teacher and student models for distillation task on all the datasets.

Dataset	# parameters Student	Teacher	%reduction
MSLR-WEB30K	0.018M	0.795M	97.8%
Yahoo LETOR	0.090M	1.372M	93.5%
Istella	0.028M	0.881M	96.8%
MSMARCO	4.4M	110.1M	96.0%

approaches. Furthermore, the results are statistically significant when measured using a paired t-test with significance level 0.05. We also observe the distilling student model from a teacher model improves over simply training student on clicks or relevance scores, illustrating the benefit of distillation. Table 6 provides details about the model size of the teacher and student model used in our experiments. The number of parameters can be used as a proxy of the inference speedup of student over teacher. Table 6 shows the number of parameters for student and teacher models for distillation task on all three datasets, and demonstrates that the student models have a significant reduction in number of parameters compared to the teacher model; thereby, leading to significant gains in inference speed. For the task of ranking loss, Table 5 shows that on MSLR-WEB30K and Yahoo LETOR datasets, RankDistil-Loss significantly outperforms other ranking losses, showing its effectiveness not just for distillation, but as a ranking loss as well. We observed that RankDistil$_c$ is either competitive or better in all our experiments. Note that these results are statistically significant when measured using paired t-test with significance level 0.05.

Recall that the model architectures for RankDistil and the baseline are exactly the same. Thus, the proposed and baseline approaches have the same inference speed. Furthermore, the training speed is also roughly similar due to the change in the distillation loss. Therefore, under the same inference latency constraints, RankDistil outperforms other distillation losses for ranking.

5 Conclusion

In this paper, we studied distillation algorithms for top-k ranking problems. We developed a novel distillation approach, RankDistil, for this setting and established statistical basis for the algorithm. The core idea behind the approach is to preserve the order of the top-k items scored by the teacher while penalizing the items ranked low by the teacher. As part of this approach, we proposed several loss functions that are amenable to efficient optimization when the number of items to rank is large. Our empirical results comparing RankDistil with popular ranking losses demonstrate the practical efficacy of RankDistil for knowledge distillation in ranking problems.

References

Ba, J. and Caruana, R. (2014). Do deep nets really need to be deep? In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 2654–2662. Curran Associates, Inc.

Breiman, L. and Shang, N. (1996). Born again trees. https://pdfs.semanticscholar.org/b6ba/5374ed8e09845996626bc62cc6d938e83fee.pdf.

Bucilă, C., Caruana, R., and Niculescu-Mizil, A. (2006). Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 535–541, New York, NY, USA. ACM.

Burges, C. J. C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. N. (2005). Learning to rank using gradient descent. In Raedt, L. D. and Wrobel, S., editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 89–96. ACM.

Cao, Z., Qin, T., Liu, T., Tsai, M., and Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In Ghahramani, Z., editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 129–136. ACM.

Chapelle, O. and Chang, Y. (2011). Yahoo! learning to rank challenge overview. In *Proceedings of the Learning to Rank Challenge*, pages 1–24.

Chen, W., Liu, T., Lan, Y., Ma, Z., and Li, H. (2009). Ranking measures and loss functions in learning to rank. In Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, pages 315–323. Curran Associates, Inc.

Crammer, K. and Singer, Y. (2001). Pranking with ranking. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 641–647. MIT Press.

Craven, M. W. and Shavlik, J. W. (1995). Extracting tree-structured representations of trained networks. In *Proceedings of the 8th International Conference on Neural Information Processing Systems*, NIPS'95, pages 24–30, Cambridge, MA, USA. MIT Press.

Czarnecki, W. M., Osindero, S., Jaderberg, M., Swirszcz, G., and Pascanu, R. (2017). Sobolev training for neural networks. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 4278–4287. Curran Associates, Inc.

Dato, D., Lucchese, C., Nardini, F. M., Orlando, S., Perego, R., Tonellotto, N., and Venturini, R. (2016). Fast ranking with additive ensembles of oblivious and non-oblivious regression trees. *ACM Trans. Inf. Syst.*

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159.

Furlanello, T., Lipton, Z. C., Tschannen, M., Itti, L., and Anandkumar, A. (2018). Born-again neural networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 1602–1611.

Furlanello, T., Zhao, J., Saxe, A. M., Itti, L., and Tjan, B. S. (2016). Active long term memory networks. *CoRR*, abs/1606.02355.

Gao, L., Dai, Z., and Callan, J. (2020). Understanding bert rankers under distillation. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*, pages 149–152.

Herbrich, R., Graepel, T., and Obermayer, K. (1999). Support vector learning for ordinal regression. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, volume 1, pages 97–102 vol.1.

Hinton, G. E., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pages 133–142. ACM.

Lee, J., Choi, M., Lee, J., and Shim, H. (2019). Collaborative distillation for top-n recommendation. In Wang, J., Shim, K., and Wu, X., editors, *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, pages 369–378. IEEE.

Lopez-Paz, D., Schölkopf, B., Bottou, L., and Vapnik, V. (2016). Unifying distillation and privileged information. In *International Conference on Learning Representations (ICLR)*.

Luce, R. D. (1959). *Individual Choice Behavior: A Theoretical analysis*. Wiley, New York, NY, USA.

Luo, T., Wang, D., Liu, R., and Pan, Y. (2015). Stochastic top-k ListNet. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 676–684, Lisbon, Portugal. Association for Computational Linguistics.

Menon, A. K., Rawat, A. S., Reddi, S. J., Kim, S., and Kumar, S. (2020). Why distillation helps: a statistical perspective. *CoRR*, abs/2005.10419.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). Ms marco: a human-generated machine reading comprehension dataset.

Pasumarthi, R. K., Bruch, S., Wang, X., Li, C., Bendersky, M., Najork, M., Pfeifer, J., Golbandi, N., Anil, R., and Wolf, S. (2019). TF-Ranking: Scalable tensorflow library for learning-to-rank. In *KDD*, pages 2970–2978.

Plackett, R. L. (1975). The analysis of permutations. *Applied Statistics*, 24(2):193–202.

Qin, T. and Liu, T.-Y. (2013). Introducing letor 4.0 datasets. *arXiv preprint arXiv:1306.2597*.

Reddi, S. J., Kale, S., Yu, F., Holtmann-Rice, D., Chen, J., and Kumar, S. (2019). Stochastic negative mining for learning with large output spaces. In *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1940–1949. PMLR.

Tang, J. and Wang, K. (2018). Ranking distillation: Learning compact ranking models with high performance for recommender system. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 2289–2298, New York, NY, USA. Association for Computing Machinery.

Tewari, A. and Chaudhuri, S. (2016). Generalization error bounds for learning to rank: Does the length of document lists matter? *CoRR*, abs/1603.01860.

Turc, I., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Well-read students learn better: The impact of student initialization on knowledge distillation. *arXiv preprint arXiv:1908.08962*.

Xia, F., Liu, T., and Li, H. (2009). Statistical consistency of top-k ranking. In Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22:*

23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada, pages 2098–2106. Curran Associates, Inc.

Xia, F., Liu, T., Wang, J., Zhang, W., and Li, H. (2008). Listwise approach to learning to rank: theory and algorithm. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 1192–1199. ACM.

Xie, Q., Hovy, E., Luong, M.-T., and Le, Q. V. (2019). Self-training with noisy student improves imagenet classification.

Xue, J., Li, J., and Gong, Y. (2013). Restructuring of deep neural network acoustic models with singular value decomposition. In *Interspeech*.