Stochastic Retrieval-Conditioned Reranking

Hamed Zamani*
University of Massachusetts Amherst
zamani@cs.umass.edu

Michael Bendersky*
Google Research
bemike@google.com

Donald Metzler
Google Research
metzler@google.com

Honglei Zhuang
Google Research
hlz@google.com

Xuanhui Wang
Google Research
xuanhui@google.com

ABSTRACT

The multi-stage cascaded architecture has been adopted by many search engines for efficient and effective retrieval. This architecture consists of a stack of retrieval and reranking models in which efficient retrieval models are followed by effective (neural) learning-to-rank models. The optimization of these learning-to-rank models is loosely connected to the early stage retrieval models. This paper draws theoretical connections between the early stage retrieval and late stage reranking models by deriving expected reranking performance conditioned on the early stage retrieval results. Our findings shed light on optimization of both retrieval and reranking models. As a result, we also introduce a novel loss function for training reranking models that leads to significant improvements on multiple public benchmarks. Our findings provide theoretical and empirical guidelines for developing multi-stage cascaded retrieval models.

CCS CONCEPTS

• **Information systems → Learning to rank**.

KEYWORDS

Multi-stage cascaded architecture; ranking optimization; learning to rank; neural information retrieval

ACM Reference Format:
Hamed Zamani, Michael Bendersky, Donald Metzler, Honglei Zhuang, and Xuanhui Wang. 2022. Stochastic Retrieval-Conditioned Reranking. In *Proceedings of the 2022 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '22), July 11–12, 2022, Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3539813.3545141

1 INTRODUCTION

Most modern search engines use the multi-stage cascaded retrieval paradigm [55] for ranking. In this paradigm, an efficient lightweight retrieval model is used to select a number of candidate documents from a large-scale collection. This retrieval model is followed by one or more reranking models that have higher computational costs but are more effective. Learning-to-rank models, including neural ranking models, are often used for implementing these late-stage reranking models. A popular assumption in the community is that the early-stage retrieval model should be optimized for recall of the top N documents, while the late-stage learning-to-rank model should be optimized for a desired user-centric precision-oriented metric, such as NDCG [5, 32, 45]. However, in contrast to this assumption, some studies on the construction of learning-to-rank test collections show that improving the retrieval recall does not always lead to better reranking performance [3]. Thus, even though learning-to-rank models have been studied extensively in the last two decades, formal models that take into account the relation between the retrieval and the reranking models are still relatively underexplored.

To this end, this paper introduces a novel formulation of reranking models in a multi-stage cascaded paradigm. Without loss of generality, we focus on two-stage retrieval models (i.e., retrieval and reranking). The proposed framework, **E**xpected reranking performance **C**onditioned on first-stage **R**etrieval (ECR), does exactly as its name implies and estimates the expected performance of a reranking model conditioned on the first-stage retrieval model through a stochastic process. Our formulation is independent of the ultimate precision-oriented ranking metric that is often desired in real-world search engines. This formulation enables us to draw connections between the performance of the retrieval and reranking models. There are three primary contributions of this work, which we describe below.

First, we study the relation between retrieval and reranking models through *theoretical* exploration. In particular, we study the performance of an *optimal* reranker in a two-stage cascaded model and theoretically quantify the impact of the retrieval stage on reranking performance.

Second, we extend our initial *theoretical* exploration to sub-optimal rerankers. We model *sub-optimal* rerankers using a noisy channel applied to an optimal reranker. With no assumption on the nature or the distribution of the noise (i.e., any arbitrary noise), we draw theoretical connections between the reranking and the retrieval performance. The outcome of our theoretical exploration challenges the popular assumption that first-stage retrieval models should optimize Recall@N. In fact, we demonstrate that this is inaccurate; instead the retrieval results should consider maximizing the precision of the result list, meaning that it should also minimize the number of non-relevant documents passed to the next stage. We also provide a realistic simulation of these sub-optimal reranking models and demonstrate that even when recall is constant, we can improve reranking performance by just focusing on precision.

---

[*] Both authors contributed equally to the paper.

ICTIR '22, July 11–12, 2022, Madrid, Spain

ACM ISBN 978-1-4503-9412-3/22/07.
https://doi.org/10.1145/3539813.3545141

Third, we demonstrate the impact of our theoretical exploration in *practice* by proposing a novel reranking optimization algorithm built on the ECR framework. We provide a robust, efficient, and effective estimation of the reranking performance conditioned on the retrieval model by viewing reranking as sampling without replacement. For this, we employ the Gumbel-Top-K trick [28] for providing differentiable estimates. Motivated by our theoretical findings mentioned above, we further extend our optimization algorithm by jointly modeling reranking and rank cutoff prediction for the retrieval model. Our empirical results on three publicly available benchmarks demonstrate that the proposed optimization algorithms significantly outperform competitive baselines.

Our contributions and findings provide practical guidelines for developing real-world multi-stage cascaded retrieval models with solid theoretical underpinnings.

2 RELATED WORK

A major inspiration for this work is the *cascade ranking model* [55], which views retrieval as a "multi-stage progressive refinement problem, where each stage considers successively richer and more complex ranking models, but over successively smaller candidate document sets". In theory, there could be an unlimited number of cascade stages. In practice, however, most existing search engines employ the *two-stage* retrieval architecture popularized by learning-to-rank approaches [31]. First, a fixed number of top candidates is retrieved for each query by a retrieval method, which operates on the inverted index (e.g., BM25), or using deep retrieval methods (e.g., DPR [26]). Then, a *(re)ranking model*, either neural or feature-based, is applied to these top candidates [14, 39, 45]. Note that in all of these works, there is an implicit assumption of *stage independence*. I.e., the rerankers are developed in isolation from the retrieval stage, and the particular methods underlying the production of the top candidates are not considered when training the reranking model.

Thus, the common intuition is that the retrieval stage should optimize for *Recall@N* [14, 32], while being bound by some cost and latency constraints [34, 55]. This is reflected in best-practices and expert advice. For example, a popular benchmarking study for TREC-COVID participants notes: "…choosing the top k documents to rerank has a large impact on end-to-end effectiveness. Reranking the top 100 seems to provide higher precision than top 1000, but the likely tradeoff is lower recall".[1] This observation is also supported by recent studies of dynamic ranking cutoffs, which suggest the sub-optimality of fixed-sized candidate lists [4, 30].

In this work, we rethink the reranking process as *sampling* from a categorical distribution of the retrieved documents, akin to the Plackett-Luce model [42]. This is similar to viewing the ranking process as inherently stochastic, as is done in some recent work. For instance, Bruch et al. [7] uses Gumbel sampling to provide more robustness for direct metric optimization, while Diaz et al. [17] use it for deriving an expected exposure evaluation metric. Roitman et al. [49] proposed a method that applies cross entropy to the permutations of the retrieval list and takes advantage of query performance prediction models for obtaining more accurate reranking models. To the best of our knowledge, our work is the first to apply the notion of ranking stochasticity in the context of the cascade ranking model.

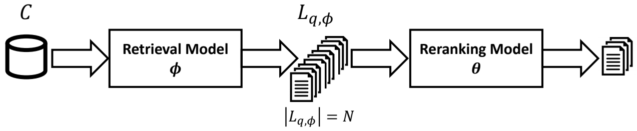

Figure 1: A two-stage cascaded retrieval model. The early-stage model, called the *Retrieval* model ϕ, efficiently retrieves N documents (denoted as $L_{q,\phi}$), followed by a late-stage model, called the *Reranking* model θ that only reranks the documents in $L_{q,\phi}$. The reranking model often focuses on improving precision-oriented metrics for the very high positions in the ranked list.

3 BACKGROUND AND NOTATION

In this section, we provide background information on the multi-stage cascaded architecture for information retrieval, and the learning-to-rank optimization used in its reranking stage. We then briefly discuss score distributions produced by several first-stage retrieval models. The notation used in this section and throughout the paper is presented in Table 1.

3.1 Multi-Stage Cascaded Retrieval

Current modern search engines often use a multi-stage cascaded architecture that includes a stack of retrieval and reranking models [6, 13, 18, 33, 55, 57]. The most common implementation of the multi-stage cascaded architecture usually involves two stages [2, 9]. As depicted in Figure 1, a lightweight scoring function ϕ (e.g., TF-IDF [50], BM25 [47], QL [43], or their combination) is applied to the entire collection C at the first-stage, which results in a large pool of N candidate documents (denoted as $L_{q,\phi}$). At this stage, existing systems mainly focus on optimizing Recall@N of the returned candidate pool and ignore their exact ordering.

At the later (second) stage, a reranking function θ is applied to the N retrieved candidates. In this stage, we generally focus on user-centric metrics that measure relevance at the very top of the ranked list, denoted as $M(q, L_{q,\phi}, k)$, where k is the ranking cutoff for the metric and $k \ll N$. NDCG@k and Precision@k are examples of such metrics. The reranking models are often implemented using learning-to-rank methods.

Increasing the number of stages [55] as well as applying multiple stages in parallel [33] have also been explored in the literature. However, in this paper, we primarily focus on a two-stage cascaded model, which is the most common implementation, and leave the extension to multiple stages to future work.

Much of the prior work [5, 32, 45] assumes that gains in Recall@N will be consistent with further gains in NDCG@k, however this is mostly based on intuition, rather than theoretical analysis. This paper closes this gap and challenges this popular belief.

3.2 Learning-to-Rank

Learning-to-rank (LTR) methods are supervised techniques heavily used as rerankers in search engines and recommender systems

---

[1] https://github.com/castorini/anserini/blob/master/docs/experiments-covid-doc2query.md

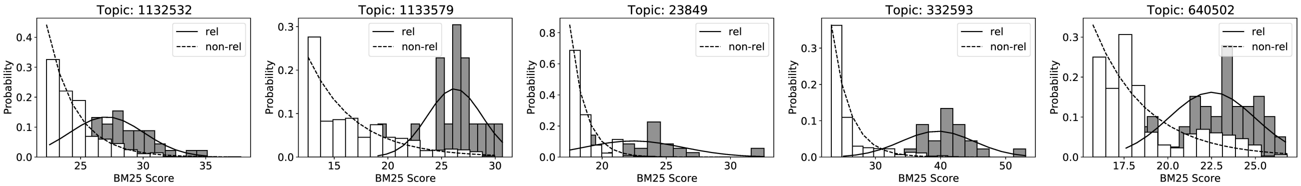

Figure 2: The score distribution of relevant (gray) and non-relevant (white) documents in the top 1000 documents returned by BM25 for five random queries from the TREC 2020 Deep Learning Track. The dashed line is a exponential distribution fit to the non-relevant document scores and the solid line is a normal distribution fit to the relevant document scores.

[9, 31]. Given a training query set $Q = \{q_1, q_2, \cdots, q_m\}$ containing m queries, the training data for LTR models can be represented as $\mathcal{T} = \{(q, D_q, R_q) : \forall q \in Q\}$, where $D_q \subseteq C$ and R_q respectively represent a set of candidate documents and relevant documents associated with the query q. Without loss of generality, we assume that relevance labels are binary (i.e., relevant or non-relevant). The LTR objective is to learn a function that produces an ordering of items in D_q for each query q so that the utility of the ordered list is maximized.

Most LTR algorithms formulate the problem as learning a ranking function to score and sort the items in a list. As such, the goal of LTR boils down to finding a parameterized ranking function $f(\cdot, \cdot; \theta)$ to minimize the empirical loss:

$$\arg\min_{\theta} \frac{1}{|Q|} \sum_{q \in Q} \mathcal{L}(f(q, D_q; \theta), R_q), \tag{1}$$

where $\mathcal{L}(\cdot, \cdot)$ is the loss function on a single list. There are three main categories of LTR loss functions: pointwise, pairwise, and listwise. LTR algorithms differ primarily in how they parameterize f and how they define $\mathcal{L}$.

A major shortcoming of existing models is how they connect LTR optimization to first-stage retrieval models. The retrieval stage is often used for shaping the document set D_q (e.g., by sampling non-relevant documents from $L_{q,\phi}$ as negative samples [37]). Once these sets are constructed for training, LTR is independent of the retrieval stage ϕ (see Equation (1)). This is, in particular, true for most recent neural ranking work, where the scores provided by the retrieval stage are either ignored [45], or heuristically combined with the reranking scores post-hoc [39].

Even though recent work by Gallagher et al. [18] studied joint optimization of multi-stage cascaded systems, they ignored the retrieval stage and solely optimized the following reranking stages jointly (they assumed more than one reranking stage). Therefore, these existing joint optimization models are also independent of the retrieval model, and are orthogonal to our work.

3.3 Retrieval Score Distributions

Modeling the score distribution produced by first-stage retrieval models has been extensively studied in the literature. It has applications to ranked list truncation, score normalization across search engines [16, 36], score thresholding for information filtering [1, 60], and query performance prediction [41, 53]. One approach to modeling score distribution is using prior relevance information. This enables us to model score distribution for both relevant and non-relevant documents. In this work, we demonstrate that modeling score distribution can also be useful in connecting the retrieval and the ranking stages, by providing a more robust estimation of document relevance to the reranker. There are multiple suggestions in prior work on modeling score distributions. For instance, Kanoulas et al. [25] suggest Gamma distribution, while Robertson et al. [48] suggest logistic distribution. In this work, we respectively use the normal and exponential distributions to describe relevant and non-relevant documents, as suggested in [1, 36, 60]. We find that these distributions fit well empirically on our datasets. To demonstrate this, Figure 2 plots the relevant and non-relevant score distributions of BM25 for five random queries sampled from the data used in the TREC 2020 Deep Learning Track. Regardless of the choices made here, it is important to note that our framework is general enough to be used with other score distributions.

Table 1: Notations frequently used throughout this paper.

Notation	Definition
C	collection
q	query
$\overline{R_q}$	a set of judged documents non-relevant to q
R_q	a set of documents relevant to q
$L_{q,\cdot}$	ranked list produced by the given model for q
$M(q, D, k)$	a ranking metric for q and list D at cutoff k
ϕ	first-stage retrieval model
θ	second-stage reranking model
γ	retrieval list truncation model
$s^{\cdot}_{qd}$	ranking score produced by the given model for q and d
N	$\lvert L_{q,\phi} \rvert$
n	number of relevant documents in $L_{q,\phi}$, i.e., $L_{q,\phi} \cap R_q$
ρ	N/n (inverse precision of the first-stage model)
$\pi_k(\cdot)$	a set of all permutations of the given list truncated at rank k
π_k	a shorthand for $\pi_k(Lq, \phi)$.

4 EXPECTED RERANKING PERFORMANCE CONDITIONED ON FIRST-STAGE RETRIEVAL

In this section, we introduce ECR, a framework for estimating performance expectation of a multi-stage cascaded system. For simplicity, assume that there are two stages: an efficient early stage model, named the *retrieval* model ϕ, followed by an effective model that reranks the output of the retrieval model, named the *reranking* model θ.

Let $M(q, D, k)$ be an evaluation metric function that takes a query q and a ranked list of documents D and computes a ranking metric with a cutoff of k. The expected performance of this ranking model for a query q would be:

$$\mathbb{E}(M(q,D,k)|\phi,\theta) = \sum_{D\in\pi(C)} p_q(D|\phi,\theta)M(q,D,k),$$

where $p_q(D|\phi,\theta)$ is the probability of a ranked list D being produced by the two-stage ranking model $\phi \rightarrow \theta$ for query q. In this equation, $\pi(C)$ is the set of all possible ranked lists. Therefore, $|\pi(C)| = |C|!$, which makes it extremely inefficient to compute. However, we know that $M(q,D,k)$ only measures the top k documents and the rest of the documents in the ranked list do not contribute to the ranking metric $M(q,D,k)$. Therefore, we can limit this computation to the top k documents in a ranked list. In addition, since θ only reranks the N documents returned by ϕ, we have $p_q(D|\phi,\theta) = 0$, if $D \not\subseteq L_{q,\phi}$. Therefore, we can reduce the computation cost of the above expectation as follows:

$$\mathbb{E}(M(q,D,k)|\phi,\theta) = \sum_{D\in\pi_k} p_q(D|\phi,\theta)M(q,D,k), \tag{2}$$

where $\pi_k = \text{perm\_cut}(L_{q,\phi}, k)$ is a set of unique ranked lists produced by computing all permutations of documents returned by the retrieval model $L_{q,\phi}$ with a ranking cutoff of k. Therefore, $|\pi_k| = {}^{N}P_k = \frac{N!}{(N-k)!}$, which is equivalent with ordered sampling of k documents without replacement from $L_{q,\phi}$. This motivates us to model reranking models via sampling without replacement. Note that the above calculation can be quite efficient for small k, but quite costly as k increases. We will provide an efficient stochastic estimation of this expectation in Section 4.2.

4.1 Theoretical Exploration of Reranking Behavior

This section provides theoretical analysis of the expected performance of multi-stage cascaded retrieval system, as described in Equation (2). We examine two cases – an optimal reranker, which always picks the relevant document, and a more realistic sub-optimal reranker based on the noisy channel model.

Optimal Reranker. The probability of selection that an optimal reranker θ_{opt} assigns to each document d_i is:

$$p_q(d_i|\phi,\theta_{\text{opt}}) \propto \begin{cases} 1 & \text{iff } d_i \in L_{q,\phi} \cap R_q \\ 0 & \text{otherwise} \end{cases},$$

where $L_{q,\phi} \cap R_q$ denotes the set of relevant documents retrieved by ϕ. The normalization factor for this probability distribution would be equal to $|L_{q,\phi} \cap R_q| = n$. We can first assume that $n \geq k$, thus the probability that the optimal reranker assigns to any arbitrary document set D with the size of k would be equal to:

$$p_q(D|\phi,\theta_{\text{opt}}) \propto \begin{cases} 1 & \text{iff } D \subseteq L_{q,\phi} \cap R_q \\ 0 & \text{otherwise.} \end{cases}.$$

Thus every document in the top k result list produced by θ_{opt} would be relevant, resulting in $M(q,D,k) = 1$, independent of the ranking metric (note that the upper-bound for most popular normalized ranking metrics is 1). Hence, the expected performance of the multi-stage system (see Equation (2)) would be equal to 1. Therefore, as long as $n \geq k$, optimal reranker produces the optimal top k metric.

On the other hand, if $n < k$, then the optimal reranker would rank all the n relevant documents first followed by $(k-n)$ non-relevant documents. In other words, $p_q(D|\phi,\theta_{\text{opt}}) \propto 1$ iff $d_i \in L_{q,\phi} \cap R_q : \forall 1 \leq i \leq n$, zero otherwise. Unlike the previous case, the expected performance of the multi-stage system depends on the ranking metric. For instance, if the ranking metric is Precision@k, then $\mathbb{E}(M(q,D,k)|\phi,\theta_{\text{opt}}) = \frac{n}{k}$.

Considering both cases ($n \geq k$ and $n < k$), Precision@k for an optimal reranker would be equal to $\min(1, \frac{n}{k})$, where n is the number of relevant documents retrieved by ϕ and k is the ranking metric cutoff. **This finding suggests that if the ultimate goal is to improve Precision@k of a multi-stage cascaded system with an *optimal* reranker, the retrieval model ϕ should maximize $\min(1, \frac{n}{k})$, instead of Recall@N (i.e., the current popular belief).** Extension of this to other metrics, e.g., NDCG@k, is trivial.

Sub-Optimal Reranker. In practice, an optimal reranker does not exist. Therefore, it is important to also provide theoretical insights into sub-optimal rerankers. Inspired by Zamani and Croft [58], we model sub-optimal rerankers using a noisy channel – an information theoretic formulation for faulty models. A noisy channel introduces noise into the optimal performance to model a sub-optimal reranker. For any arbitrary noise distribution, the probability of selection that a sub-optimal reranker $\theta_{\text{sub-opt}}$ assigns to each document d_i is:

$$p_q(d_i|\phi,\theta_{\text{sub-opt}}) \propto \begin{cases} 1-\epsilon_i & \text{iff } d_i \in L_{q,\phi} \cap R_q \\ \epsilon_i & \text{otherwise} \end{cases}, \tag{3}$$

where each $0 \leq \epsilon_i \leq 1$ is the noise probability of the sub-optimal reranker $\theta_{\text{sub-opt}}$ for the i^{th} document. Since $\epsilon_i = 0.5$ corresponds to a random reranker, it is expected that $\epsilon_i < 0.5$. Thus, the normalization factor for this probability distribution is equal to $\sum_{d_i\in L_{q,\phi}\cap R_q} (1-\epsilon_i) + \sum_{d_i\in L_{q,\phi}\setminus R_q} \epsilon_i$. Therefore, using sampling without replacement (see more on this in Section 4.2), we can also compute $p_q(D|\phi,\theta_{\text{sub-opt}})$ for every ranked list $D \in \pi_k$. Through Equation (2), the expected performance of a sub-optimal model can be computed. However, the obtained equation is complicated for a general case. For illustration, let us assume $k = 1$, meaning that we care solely about the relevance of the first retrieved document (e.g., Precision@1). In this case, expected performance of a sub-optimal reranker would be equal to:

$$\begin{aligned} &\mathbb{E}(M(q,D,k=1)|\phi,\theta_{\text{sub-opt}}) \\ &\quad= \sum_{i=1}^{N} p_q(d_i|\phi,\theta_{\text{sub-opt}})M(q,D,k=1) \\ &\quad= \sum_{d_i\in L_{q,\phi}\cap R_q} \frac{1-\epsilon_i}{\sum_{d_j\in L_{q,\phi}\cap R_q}(1-\epsilon_j) + \sum_{d_j\in L_{q,\phi}\setminus R_q}\epsilon_j} \\ &\quad= \frac{\sum_{d_i\in L_{q,\phi}\cap R_q}(1-\epsilon_i)}{\sum_{d_j\in L_{q,\phi}\cap R_q}(1-\epsilon_j) + \sum_{d_j\in L_{q,\phi}\setminus R_q}\epsilon_j} \\ &\quad= \frac{n(1-\epsilon_+)}{n(1-\epsilon_+) + (N-n)\epsilon_-} \\ &\quad= \frac{(1-\epsilon_+)}{(1-\epsilon_+) + (\rho-1)\epsilon_-}, \end{aligned} \tag{4}$$

where ϵ_+ and ϵ_- denote the average noise probability for relevant documents (i.e., $d_i \in L_{q,\phi} \cap R_q$) and non-relevant documents (i.e., $d_i \in L_{q,\phi} \setminus R_q$), respectively. In the last step, $\rho := \frac{N}{n}$. This derivation is true for the majority of ranking metrics, including Precision, Reciprocal Rank, DCG, NDCG, and Hit Ratio. In this derivation, ϵ_+ and ϵ_- solely depend on the reranker's quality (more accurate rerankers would result in smaller ϵ_+ and ϵ_-). On the other hand, ρ solely depends on the retrieval quality by ϕ. **This theoretical finding suggests that if the ultimate goal is to improve the metrics for the first ranked document produced by a *sub-optimal* reranker, the retrieval model ϕ should minimize $\rho = \frac{N}{n}$, instead of Recall@N (i.e., the current popular belief).** Note that ρ is equal to the inverse of precision. Therefore, the retrieval model should maximize precision of the top N documents. This suggests that withholding non-relevant documents from $L_{q,\phi}$ is as important as maximizing the number of relevant documents. One approach for withholding non-relevant documents would be truncating the retrieval list $L_{q,\phi}$ through cutoff prediction. This observation motivates us to jointly model reranking and cutoff prediction of the first-stage retrieval list in Section 4.3.

Relationship between Recall and ρ. As mentioned earlier, $1/\rho$ is equal to the precision of the first-stage retrieval model. If N (the number of documents returned by the retrieval model) is constant, then increasing Recall@N would translate into increasing $1/\rho$. However, by varying N we observe that $1/\rho$ varies even if Recall@N does not change. The simulations presented in Section 5.3 demonstrate how ρ values can impact reranking performance even if Recall@N of the retrieval model does not change.

As we will see in Equation (6), the variable ρ also appears in estimating the ranking metric $M(q, D, k)$ for the reranker. This is independent of k (the ranking cutoff for the evaluation metric). Therefore, the variable ρ also plays a key role in our optimization presented in the following subsection.

4.2 ECR Optimization for Reranking Models

Building upon the theoretical analysis of the impact of the retrieval stage on the reranking performance, this section introduces a novel practical optimization approach for LTR rerankers in multi-stage cascaded systems. Given a training query set Q containing independent and identically distributed queries, ECR optimization suggests the following based on empirical expectation:

$$\begin{aligned}\theta^* &= \arg\max_{\theta} \mathbb{E}_Q \left[\mathbb{E}(M(q, D, k)|\phi, \theta)\right] \\ &= \arg\max_{\theta} \frac{1}{|Q|} \sum_{q\in Q} \mathbb{E}(M(q, D, k)|\phi, \theta) \\ &= \arg\max_{\theta} \frac{1}{|Q|} \sum_{q\in Q} \sum_{D\in\pi_k} p_q(D|\phi, \theta) M(q, D, k) \qquad (5)\end{aligned}$$

The derivation above follows from the Law of Iterated Expectation and Equation (2). The derivation assumes that all queries in the query set are uniformly weighted. As we will see in the following, we obtain a robust estimation of $M(q, D, k)$ using the retrieval scores produced by ϕ and the relevance judgments. Thus, $M(q, D, k)$ is independent of the reranking model and can be computed offline. On the other hand, $p_q(D|\phi, \theta)$ is the probability that a reranker assigns to the given ranked list D.

In this particular case, it is useful to consider reranking as a process of iteratively selecting items from the retrieved list of candidates $L_{q,\phi}$ based on their scores. As rerankers are imperfect in practice, we can model this process via *sampling without replacement* from the Plackett-Luce distribution [42]. For estimating $p_q(D|\phi, \theta)$, we first compute the sampling probability of each individual document that appears in $L_{q,\phi}$ using the softmax operator:

$$p_q(d|\phi, \theta) = \frac{\exp\,(s^{\theta}_{qd})}{\sum_{d'\in L_{q,\phi}} \exp\,(s^{\theta}_{qd'})}$$

For every $D \in \pi_k$, the probability of sampling without replacement from the above distribution can be computed using an incremental approach as follows:

$$p_q(D|\phi, \theta) = \prod_{i=1}^{k} \frac{p_q(d_i|\phi, \theta)}{1 - \sum_{j=1}^{i-1} p_q(d_j|\phi, \theta)}$$

Intuitively, this approach takes samples one by one and renormalizes the distribution once each sample is selected. The Placket-Luce distribution is a common choice for modeling ranking processes. In fact, several popular learning-to-rank methods, such as ListNet [8], ListMLE [56], and Softmax Cross-Entropy [40] are based on it. However, directly sampling from the Plackett-Luce distribution is computationally expensive, as it requires instantiating the set of all possible permutations π_k. This may not be tractable, especially for larger lists. Furthermore, this iterative process of document sampling is non-differentiable, and thus cannot be simply used in gradient descent-based optimization approaches. To address both of these problems, Kool et al. [28, 29] recently introduced Ancestral Gumbel-Top-k sampling. This approach creates a tree over all items in the sampling set and extends the Gumbel-Softmax sampling approach [35] to sampling without replacement. According to [28], independently perturbing each individual document score with Gumbel noise and picking the top k documents with the largest perturbed values will generate a valid sample from the Plackett-Luce distribution. Gumbel perturbation itself can be done efficiently by simply drawing a sample $U \sim \text{Uniform}(0, 1)$, as $\text{Gumbel}(0, \beta) \sim -\beta \log(-\log(U))$ [35].

Algorithm 1 demonstrates how this Gumbel-Top-k sampling technique can be integrated in a gradient descent-based learning-to-rank method to generate samples to approximate Equation (5). This algorithm is run iteratively at each training epoch, such that new samples are generated using the updated model weights θ.

Note that Algorithm 1 can be viewed as a generalization of the stochastic reranking approach first introduced by Bruch et al. [7]. In the stochastic reranking setting, we also generate Gumbel-perturbed score samples at each training epoch. However, in that approach, the samples can be viewed as uniformly weighted, whereas we introduce an explicit dependence on the retrieval stage by attaching a metric estimate $M(q, D, k)$ to each sample D. We evaluate the empirical benefits of this weighting in Section 5.3.

Robust Estimation of $M(q, D, k)$. One can simply compute this evaluation metric using relevance judgments. This is only possible when complete relevance judgments are available for training

Algorithm 1 Gumbel-Top-K sampling for ECR optimization.

```
Input (a) a set of queries $Q$; (b) a list of candidate documents $L_{q,\phi}$; $\forall q \in Q$; (c) current ranking model parameters $\theta$, which can be used to derive log-probabilities $s^{\theta}_{qd}$; $\forall d \in L_{q,\phi}$
Output (a) $S$ samples with Gumbel score perturbation drawn from $L_{q,\phi}$; $\forall q \in Q$; (b) a metric estimate $M(q, D, k)$ for each sample.
for $q \in Q$ do
    for $i \in [1, 2, \dots, S]$ do
        $U_d \sim \text{Uniform}(0, 1)$, $G_d = -\beta \log(-\log(U_d))$; $\forall d \in L_{q,\phi}$
        $\tilde{p}_q(d|\phi, \theta) = \frac{\exp(s^{\theta}_{qd} + G_d)}{\sum_{d' \in L_{q,\phi}} \exp(s^{\theta}_{qd'} + G_{d'})}$; $\forall d \in L_{q,\phi}$
        $\tilde{\mathbf{p}}_q \leftarrow [\tilde{p}_q(d_1|\phi, \theta), \dots, \tilde{p}_q(d_N|\phi, \theta)]$
        yield $(\tilde{\mathbf{p}}_q, M(q, \tilde{\mathbf{p}}_q, k))$
    end for
end for
```

the model. Given the large number of queries required for training most of the current state-of-the-art neural ranking models [12], it is impractical to have complete relevance judgment for a large number of queries. Therefore, we propose a more robust estimation for this metric function based on the probability of relevance for each query-document pair. Let $R = \{0, 1\}$ be a binary random variable representing the notion of relevance. Hence, $p(R = 1|q, d)$ denotes the probability of relevance of document d to query q. Using this probability we estimate popular precision-oriented ranking metrics for a single query q for the top k retrieved documents. These metrics include:

- **Precision.** The ratio of relevant documents in the top k retrieved documents.
- **Reciprocal Rank (RR).** The multiplicative inverse of the rank of the first relevant document.
- **Discounted Cumulative Gain (DCG)** [24]. The relevance gain obtained by a ranked list of documents discounted by their positions in the ranked list.
- **Hit Ratio.** The ratio of ranked lists across a query set that contain at least one relevant document (or attract click from the user). This metric is quite popular for evaluating recommender systems [52]. Hit Ratio for a single query (or user) is binary (either containing a relevant document or not).

The estimation of these metrics are reported in Table 2. For all the metrics, we assume binary relevance. Note that DCG is often normalized (i.e., NDCG) by the ideal DCG value. Given the assumption that $n \geq k$ (which is reasonable for small values of k), ideal DCG for cutoff k with the binary relevance assumption would be a constant value.

Estimation of popular precision-oriented ranking metrics therefore boils down to an accurate and robust estimation of $p(R = 1|q, d)$ for every document in $L_{q,\phi}$. Given Bayes' rule and the Law of Total Probability, we have:

$$\begin{aligned} p(R = 1|q, d) &= \frac{p(d|q, R = 1)p(R = 1|q)}{p(d|q)} \\ &= \frac{p(d|q, R = 1)p(R = 1|q)}{p(d|q, R = 1)p(R = 1|q) + p(d|q, R = 0)p(R = 0|q)} \end{aligned}$$

Table 2: Estimation of popular precision-oriented ranking metrics for a given query q and ranked list D. In these equations, the ranking cutoff is k and d_i denotes the i^{th} document in the ranked list D. For all metrics we assume relevance is a binary notion.

Metric	Estimation of M(q, D, k)
Precision	$\sum_{i=1}^{k} p(R = 1\|q, d_i)/k$
Reciprocal Rank	$\sum_{i=1}^{k} \frac{1}{i} p(R = 1\|q, d_i) \prod_{j=1}^{i-1} (1 - p(R = 1\|q, d_j))$
DCG	$\sum_{i=1}^{k} (p(R = 1\|q, d_i)/\log(i + 1))$
Hit Ratio	$1 - \prod_{i=1}^{k} (1 - p(R = 1\|q, d_i))$

Since this probability of relevance is computed for the reranking model and $d \in L_{q,\phi}$, we can estimate $p(R = 1|q) = \frac{n}{N}$, where n denotes the number of relevant documents in $L_{q,\phi}$, i.e., the top N documents returned by ϕ. Defining $\rho = \frac{N}{n}$ as we did previously, we can rewrite the above equation as follows:

$$p(R = 1|q, d) = \frac{p(d|q, R = 1)}{p(d|q, R = 1) + (\rho - 1)p(d|q, R = 0)} \tag{6}$$

The probability $p(d|q, R = 1)$ denotes the probability of d being generated from the relevance distribution for query q. As discussed in Section 3.3, this distribution can be modeled by a normal distribution (denoted by $\mathcal{N}(\cdot, \cdot)$) over the retrieval scores. In addition, $p(d|q, R = 0)$ can be estimated using an exponential distribution (denoted by $\text{Exp}(\cdot)$). Therefore, we estimate the above probability as follows:

$$p(R = 1|q, d) = \frac{p(s^{\phi}_{qd} \sim \mathcal{N}(\mu, \sigma))}{p(s^{\phi}_{qd} \sim \mathcal{N}(\mu_q, \sigma_q)) + (\rho - 1)p(s^{\phi}_{qd} \sim \text{Exp}(\lambda_q))}, \tag{7}$$

where s^{ϕ}_{qd} denotes the retrieval score for the query q and document d produced by ϕ. Therefore, we need to fit the two distributions $\mathcal{N}(\mu_q, \sigma_q)$ and $\text{Exp}(\lambda_q)$ to the relevant and non-relevant document score distributions by estimating the three query-dependent parameters μ_q, σ_q, and λ_q. They can be estimated by maximizing the log-likelihood of generating the observed retrieval scores for labeled documents in the judgment set:

$$\mu_q, \sigma_q = \arg\max_{\mu,\sigma} \sum_{d \in R_q} \log(p(s^{\phi}_{qd}|\mathcal{N}(\mu, \sigma))),$$

$$\lambda_q = \arg\max_{\lambda} \sum_{d \in \overline{R_q}} \log(p(s^{\phi}_{qd}|\text{Exp}(\lambda))),$$

These two optimization processes can be trivially solved using the Lagrange multiplier approach; see previous work on modeling score distributions, e.g., [1, 36]. Note that these optimizations are independent of the reranking model θ and thus can be done offline.

Extension to Multiple Reranking Stages. So far we solely focus on two-stage retrieval models. As suggested by Wang et al. [55], the multi-stage cascaded architecture may involve more than one reranking stage. Let assume that the retrieval model ϕ is followed by

m rerankers $\phi_1, \phi_2, \cdots, \phi_m$ in that order. These m rerankers can be models using Markov chain. Thus, inspired by Gallagher et al. [18], we can extend the ECR optimization to multiple rerankers using the backpropagation algorithm. With the assumption that all reranking stages are differentiable and they are parameterized independently, each reranker θ_i can be modeled as a sampling without replacement process. Thus, Algorithm 1 can be applied for every reranker θ_i. Since this algorithm provide a differentiable approximation of sampling without replacement, the whole process for a stack of rerankers would be differentiable. For more information on joint optimization of multiple rerankers using backpropagation, we refer the reader to [18].

4.3 ECR Optimization for Joint Reranking and Retrieval List Truncation

According to the theoretical findings presented in Section 4.1, truncating the retrieval result list to avoid including a large number of non-relevant documents in the lower ranks can improve reranking quality. Based on this observation, we extend the ECR reranking optimization approach to jointly optimize retrieval list truncation and reranking quality. Let γ denote a ranking truncation model that produces a probability distribution over all rank positions (i.e., $p_{\text{cutoff}}(i|\phi, \theta, \gamma)$) where a higher probability represents a higher chance of truncation. Given this, we rewrite Equation (5) as follows:

$$\begin{aligned}
\theta^*, \gamma^* &= \arg\max_{\theta,\gamma} \mathbb{E}_Q \left[\mathbb{E}(M(q, D, k)|\phi, \theta, \gamma)\right] \\
&= \arg\max_{\theta,\gamma} \frac{1}{|Q|} \sum_{q\in Q} \mathbb{E}(M(q, D, k)|\phi, \theta, \gamma) \\
&= \arg\max_{\theta,\gamma} \frac{1}{|Q|} \sum_{q\in Q} \sum_{D\in\pi_k} p_q(D|\phi, \theta, \gamma) M(q, D, k) \\
&= \arg\max_{\theta,\gamma} \frac{1}{|Q|} \sum_{q\in Q} \sum_{i=k}^{N} [p_{\text{cutoff}}(i|\phi, \theta, \gamma) \cdot \\
&\qquad \sum_{D\in\pi_k(L_{q,\phi}[:i])} p_q(D|\phi, \theta) M(q, D, k)]
\end{aligned} \tag{8}$$

where $L_{q,\phi}[:i]$ denotes the first i documents returned by the retrieval model ϕ. Note that considering our ultimate ranking metric M, the minimum cutoff rank is k. We can calculate components $p_q(D|\phi, \theta)$ and $M(q, D, k)$ using the equations presented in Section 4.2. The remaining probability $p_{\text{cutoff}}(i|\phi, \theta, \gamma)$ can be estimated using a list truncation model. We use the Choppy architecture [4] but optimize it jointly with our rerankers. The Choppy architecture is a Transformer-based [54] ranked list truncation model that takes the scores produced by ϕ as input.

5 EXPERIMENTS

We conduct extensive experiments to evaluate the proposed approaches. Our experiments involve simulation of sub-optimal rerankers as well as evaluation on diverse standard retrieval benchmarks.

5.1 Data and Experimental Setup

In our experiments, we focus on two-stage retrieval models. The first-stage model is implemented using BM25 [47]. The second-stage model is a standard BERT [15] reranking model first proposed by Nogueira and Cho [38]. This model takes `[CLS] query tokens [SEP] doc tokens [SEP]` as input and applies a single fully-connected layer to the CLS representation of the last layer in order to produce a real-valued relevance score. We use BERT-base that is fine-tuned on the MS MARCO training set (503K training queries) in all experiments.[2] We truncate the input sequences that are longer than 512 tokens due to BERT's input length limitations. Neural ranking architectures that handle long documents, such as TKL [23] and IDCM [22], can simply use the proposed optimization approach. Note that there exist some models with different architectures or larger parameter sets that slightly outperform BERT reranking models, such as those based on BERT-large or T5. However, the focus of our experiments is on optimization of these models which can be applied to all of these alternatives. For this reason, we focus on this standard BERT reranking model architecture and train it using a wide range of optimization approaches. We use Adam optimizer [27] with a learning rate of 7×10^{-6}. We employ early stopping, based on the best NDCG@10 value on the validation set. We set the number of perturbation samples S to 128.

For evaluation, we use the following diverse benchmarks:

TREC DL 2020: This data was used for the TREC Deep Learning Track in 2020 [11]. We focus on the passage retrieval task that consists of 54 queries and a collection of 8.8 million passages. We use the TREC DL 2019 dataset [10] for training (43 queries).

TREC Robust 2004: This data was used for the TREC Robust Track in 2004. The collection consists of 528K news articles from multiple news agencies. It consists of 249 queries. Due to the lack of training set for this dataset, we use 5-fold cross-validation over the query set (i.e., each fold consists of about 50 queries). There are an average of 70 relevant documents per query in this dataset.

ANTIQUE: This is a passage retrieval dataset for non-factoid questions developed by Hashemi et al. [19]. The questions are sampled from real user questions in the Yahoo! Webscope L6 dataset, a community question answering dataset. For this benchmark, we train the model on the ANTIQUE's training set (2426 queries) and evaluate on its test set containing 200 queries. It contains 8.5 relevant document per query on average. The relevance judgments in ANTIQUE are constructed using the standard depth-k pooling techniques used by TREC.

As mentioned above, note that for all three benchmarks, the models were trained on the MS MARCO training set before fine-tuning on the corresponding small training sets.

5.2 Evaluation Metrics

The focus of this work is on precision-oriented metrics for the very top ranks in the result list. Hence, we use NDCG [24] with different ranking cutoffs (1, 3, and 10). Furthermore, we use two-tailed paired t-tests with Bonferroni correction for identifying statistically significant performance differences ($p\_value < 0.05$).

---

[2] https://github.com/nyu-dl/dl4marco-bert

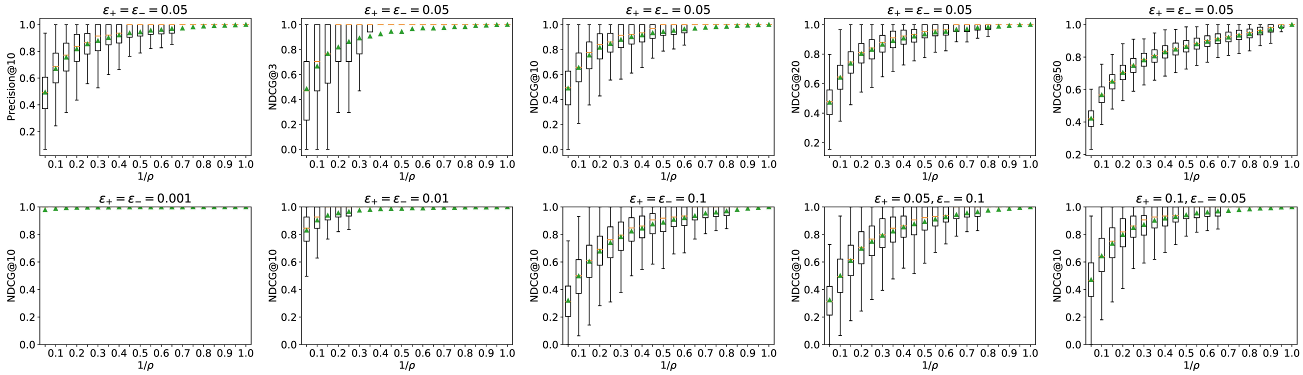

Figure 3: Simulating a sub-optimal reranker's performance for different evaluation metrics at different ranking cutoffs (top row) and in terms of NDCG@10 for different values of ϵ_+ and ϵ_- (bottom row). In this simulation, we kept the recall of the retrieval model fixed (set to 0.5) to demonstrate that by increasing $1/\rho$, the reranking performance can be improved independent of the recall value.

Table 3: Performance of two-stage retrieval models with different optimization approaches, in terms of NDCG@k. The superscripts * and † denote statistically significant improvements compared to all the baselines and ECR, respectively. Highest value in each column is bold-faced (except for the Oracle).

	Reranking	TREC DL 2020			TREC Robust 2004			ANTIQUE		
Retrieval Model	**Optimization**	**@1**	**@3**	**@10**	**@1**	**@3**	**@10**	**@1**	**@3**	**@10**
BM25	–	0.5432	0.5256	0.4980	0.5341	0.4908	0.4369	0.4411	0.4237	0.4334
$\text{BM25} \xrightarrow{N=1000} \text{BERT}$	Cross-entropy (pointwise)	0.6728	0.6464	0.6149	0.5778	0.5372	0.4732	0.7126	0.6570	0.6423
	Cross-entropy (pairwise)	0.6773	0.6497	0.6168	0.5975	0.5420	0.4795	0.7348	0.6943	0.6776
	Hinge loss (pairwise)	0.6766	0.6474	0.6192	0.5971	0.5346	0.4719	0.7456	0.6917	0.6791
	ApproxNDCG (listwise)	0.6881	0.6511	0.6288	0.6035	0.5587	0.4872	0.7589	0.6970	0.6841
	λMART (listwise)	0.6853	0.6497	0.6208	0.6055	0.5513	0.4826	0.7506	0.6957	0.6815
	$\mathbb{E}$(ApproxNDCG)[7]	0.7076	0.6634	0.6438	0.6115	0.5601	0.4948	0.7767	0.7234	0.7071
	$\mathbb{E}$(λMART)[7]	0.6915	0.6605	0.6353	0.6150	0.5576	0.4976	0.7565	0.7020	0.6917
	ECR	0.7160^{*}	0.6774^{*}	0.6523^{*}	0.6221	0.5647^{*}	0.4981	0.8044^{*}	0.7423^{*}	0.7336^{*}
$\text{BM25} \xrightarrow{\text{Oracle}} \text{BERT}$	ECR	$0.7604^{*\dagger}$	$0.7470^{*\dagger}$	$0.7234^{*\dagger}$	$0.7303^{*\dagger}$	$0.6286^{*\dagger}$	$0.5548^{*\dagger}$	$0.9613^{*\dagger}$	$0.8642^{*\dagger}$	$0.8158^{*\dagger}$
$\text{BM25} \xrightarrow{\text{Choppy}} \text{BERT}$	ECR	0.6713	0.6485	0.6212	0.6012	0.5406	0.4312	0.7980	0.6982	0.6777
$\text{BM25} \xrightarrow{\text{Choppy}} \text{BERT}$	Joint ECR	**0.7184***	**0.6809***	**0.6593***	**0.6365***†	**0.5717***	**0.5069***	**0.8123***†	**0.7639***†	**0.7591***†

5.3 Experimental Results

We conducted several experiments to empirically study the five following research questions.

RQ1. How does ρ impact the performance of a sub-optimal reranker? As mentioned in Section 4.1, there is a theoretical connection between the performance of a sub-optimal reranker and ρ. We formally derive this connection in Equation (4) for metrics when the ranking cutoff $k = 1$. To complement our theoretical derivation, we demonstrate the impact of ρ on metrics for deeper ranked lists through a number of simulations. To this aim, we generate 1000 samples of the top k retrieved documents from the sub-optimal reranking distribution mentioned in Equation (3).

The top row of Figure 3 shows the simulation results for $\epsilon_+ = \epsilon_- = 0.05$, $N = 2000$, and $n = 50$ for different ranking metrics, where the recall is fixed and set to 0.5. Having fixed recall enables us to observe the true impact of ρ on the reranking performance. The first plot (left) in the top row of Figure 3 shows the precision metric while the others show NDCG. The aim is to show that the observations hold for both ranking metrics. Overall, the top row shows that ρ values have more impact on NDCG for deeper rankings (higher k). We make similar observations for Precision, but the results are omitted due to space limitations.

We repeat this simulation by varying the ϵ_+ and ϵ_- values. The results are presented in the bottom row of Figure 3. It shows that the higher these values (the less accurate the reranking model) the more impact ρ has on the ultimate NDCG@10 values. The left plot ($\epsilon_+ = \epsilon_- = 0.001$) demonstrates the performance for an almost-optimal reranker. This observation is inline with our theoretical findings for an optimal reranker.

RQ2. How does the proposed ECR optimization algorithm work compared to existing optimization approaches? To address this research question, we consider a number of optimization algorithms for reranking the results produced by BM25 as the first-stage retrieval model. In this set of experiments, we rerank the top

1000 documents returned by BM25 using a BERT model as described in Section 5.1. This is a popular and competitive approach for retrieval. Therefore, the architecture of the reranking model is fixed, while we use various optimization approaches from popular pointwise to pairwise and to listwise approaches for training reranking models. We also include two stochastic optimization approaches (i.e., $\mathbb{E}$(ApproxNDCG) and $\mathbb{E}(\lambda\text{MART})$) recently proposed by Bruch et al. [7]. Note that this paper does not focus on dense retrieval or knowledge distillation, so methods like [20, 21, 44, 46, 51, 59] are out of the scope of this work. According to results presented in Table 3, all reranking methods improve the retrieval model's performance in terms of NDCG metrics (@1, @3, and @10), which is expected. We also show that listwise approaches generally produce better results than pointwise and pairwise models. Some additional improvements can be observed by considering the stochastic optimization solutions. That being said, the proposed optimization approach (ECR) outperforms all the baselines across all three datasets. The improvements are statistically significant in nearly all cases. We observe substantially larger improvements on the ANTIQUE dataset. The reason is due to the larger training set that ANTIQUE provides for training ECR. The training set of ANTIQUE is 10 times larger than the training set of the other two datasets (recall that it is different from the initial warm-up training on MS MARCO that is used for all three datasets; see Section 5.1 for more details).

RQ3. How do our theoretical findings about retrieval stage convey into practice? In Section 4.1, we showed that optimizing the first stage retrieval for recall is sub-optimal, and instead we suggested to minimize ρ if there are more than k retrieved relevant documents. Addressing this research question empirically validates this theoretical finding. To do so, we consider an Oracle ranked list truncation model that finds the optimal cutoff of the ranked list produced by BM25 for the BERT reranker. The maximum cutoff value is set to 1000. Therefore, the recall for a fixed ranked-list with $N = 1000$ documents is greater than or equal to the recall of the Oracle, denoted BM25 $\xrightarrow{\text{Oracle}}$ BERT. As reported in Table 3, this Oracle model substantially outperforms BM25 $\xrightarrow{N=1000}$ BERT with ECR optimization. This suggests that even though the reranking model and its optimization are the same for both approaches, higher retrieval recall (obtained by $N = 1000$) leads to worse performance. Thus, optimizing recall for the retrieval stage is not desired, which validates our theoretical findings. This is due to the number of non-relevant documents that are included by this fixed cutoff value. The next research question goes beyond Oracle models.

RQ4. How does the proposed joint reranking and rank truncation model perform compared to the baselines? According to the results presented in Table 3, the proposed Joint ECR model that jointly optimizes for cutoff prediction and reranking outperforms the ECR model (and thus all the reranking baselines discussed in response to RQ1). Improvements obtained over all the baselines are statistically significant on all datasets. However, comparing ECR and Joint ECR suggests that significant differences are not generally observed on TREC DL and Robust datasets. One possible reason for this observation is that the number of relevant documents in these two datasets is substantially higher than in ANTIQUE, and thus increase in ρ by the rank truncation is more difficult.

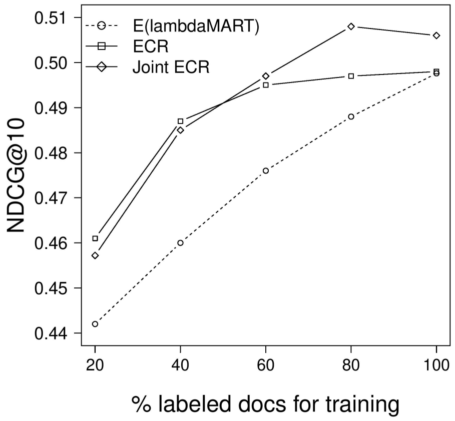

Figure 4: Robustness of $\mathbb{E}(\lambda\text{MART})$, ECR, and Joint ECR to incomplete relevance annotation on TREC Robust 2004 dataset, in terms of NDCG@10.

Another natural baseline for our joint model is a ranked truncation model followed by the reranking model that are trained independently. Similar to our approach, we used the Choppy model [4], which is a state-of-the-art ranked list truncation approach. According to the results, independent training of ranked list truncation actually hurts the model performance. The model that reranks all the top 1000 documents returned by BM25 produces a better result. Consequently, this independent training performs significantly worse than the joint optimization model.

Comparing Joint ECR and the Oracle rank truncation model suggests that there is still a large room for improvement, and more accurate modeling of rank truncation is likely to lead to further improvements.

RQ5. How robust is the proposed model to incomplete relevance annotation? As mentioned in Section 4.2, the proposed calculation of $M(q, D, k)$ introduces a robust estimation for metrics. In this research question, we empirically evaluate the robustness of the proposed solution. To this aim, we compared ECR and Joint ECR to our most effective baseline in TREC Robust 2004 in terms of NDCG@10, i.e., $\mathbb{E}(\lambda\text{MART})$. For the sake of space, we only report NDCG@10 on the TREC Robust 2004 dataset for addressing this research question. The rational for selecting this dataset is that TREC Robust 2004 has the highest number of relevant documents per query. Thus, it is the most appropriate baseline for evaluating robustness to incomplete relevance annotation by randomly withholding some of the relevance documents. The results are presented in Figure 4. The x-axis represents the percentage of labeled documents used for training the models. We observe larger improvements compared to the baseline model when we use partial relevance annotations for training. According to the curves, both ECR and Joint ECR are more robust to incomplete relevance annotations than the baseline model.

Ablation Study. The majority of cases for ablation study has already been discussed (e.g., ECR vs. Joint ECR). In this last set of

Table 4: Evaluating the impact of proposed estimation method for $M(q, D, k)$ on the ANTIQUE dataset, in terms of NDCG@k. The superscript * denotes statistically significant improvement compared to the baseline model that directly computes NDCG for modeling M.

Method - $M(q, D, k)$	@1	@3	@10
ECR- NDCG	0.7814	0.7195	0.7143
ECR- Ours	0.8044^*	0.7423^*	0.7336^*
Joint ECR- NDCG	0.7881	0.7240	0.7207
Joint ECR- Ours	0.8123^*	0.7639^*	0.7591^*

experiments, we demonstrate the impact of the proposed robust estimation for $M(q, D, k)$. A simple approach would be using the target metric formula for computing M. We replace the function M in our model with NDCG formula and report the results on our dataset with the largest training set (i.e., ANTIQUE) in Table 4. It shows that the proposed smooth estimation of NDCG (i.e., Ours) leads to better reranking quality than the exact calculation. We observe a similar behavior on other datasets too.

6 CONCLUSIONS

In this paper, we introduced ECR, a framework for multi-stage cascaded retrieval that models the performance of reranking models conditioned on the first-stage retrieval model. Through theoretical investigation, we demonstrated that the popular belief of "optimizing Recall@N for the first-stage retrieval model" is not accurate. By theoretical modeling of both optimal and sub-optimal reranking models, we showed that including many non-relevant documents in the reranking set can hurt the retrieval performance, so we should not aim for high recall. Instead, we should aim for high precision as long as sufficient relevant documents are retrieved. Building on the proposed ECR framework, we introduced a novel optimization algorithms based on stochastic modeling of sampling without replacement. We further extended the proposed algorithm and jointly modeled reranking and truncation of the retrieval results. Extensive experiments on three diverse datasets suggest that the proposed solutions are effective in practice and our models significantly outperform the baseline optimization methods.

ACKNOWLEDGMENTS

This research was supported in part by the Google Visiting Scholar program and in part by the Center for Intelligent Information Retrieval. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Avi Arampatzis and André van Hameran. The score-distributional threshold optimization for adaptive binary classification tasks. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 285–293, 2001.

[2] Nima Asadi (Sebastian Bruch) and Jimmy Lin. Fast candidate generation for two-phase document ranking: Postings list intersection with bloom filters. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2419–2422, 2012.

[3] Javed A Aslam, Evangelos Kanoulas, Virgil Pavlu, Stefan Savev, and Emine Yilmaz. Document selection methodologies for efficient and effective learning-to-rank. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 468–475, 2009.

[4] Dara Bahri, Yi Tay, Che Zheng, Don Metzler, and Andrew Tomkins. Choppy: Cut transformers for ranked list truncation. 2020.

[5] Michael Bendersky, Honglei Zhuang, Ji Ma, Shuguang Han, Keith Hall, and Ryan McDonald. Rrf102: Meeting the trec-covid challenge with a 100+ runs ensemble, 2020.

[6] Andrei Z. Broder, David Carmel, Michael Herscovici, Aya Soffer, and Jason Zien. Efficient query evaluation using a two-level retrieval process. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, page 426–434, New York, NY, USA, 2003. Association for Computing Machinery.

[7] Sebastian Bruch, Shuguang Han, Michael Bendersky, and Marc Najork. A stochastic treatment of learning to rank scoring functions. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, page 61–69, New York, NY, USA, 2020. Association for Computing Machinery.

[8] 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 the 24th international conference on Machine learning*, pages 129–136, 2007.

[9] Olivier Chapelle and Yi Chang. Yahoo! learning to rank challenge overview. In *Proceedings of the learning to rank challenge*, pages 1–24. PMLR, 2011.

[10] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. Overview of the trec 2019 deep learning track. In *TREC*, 2019.

[11] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. Overview of the trec 2020 deep learning track. In *TREC*, 2020.

[12] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. Ms marco: Benchmarking ranking models in the large-data regime. In *Proceedings of the 44th international ACM SIGIR conference on Research & development in information retrieval*. ACM, April 2021.

[13] Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines: Information Retrieval in Practice.* Addison-Wesley Publishing Company, USA, 1st edition, 2009.

[14] Van Dang, Michael Bendersky, and W Bruce Croft. Two-stage learning to rank for information retrieval. In *European Conference on Information Retrieval*, pages 423–434. Springer, 2013.

[15] J. Devlin, M. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*, 2019.

[16] Fernando Diaz. Regularizing ad hoc retrieval scores. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, page 672–679, New York, NY, USA, 2005. Association for Computing Machinery.

[17] Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. Evaluating stochastic rankings with expected exposure. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, page 275–284, New York, NY, USA, 2020. Association for Computing Machinery.

[18] Luke Gallagher, Ruey-Cheng Chen, Roi Blanco, and J. Shane Culpepper. Joint optimization of cascade ranking models. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, page 15–23, New York, NY, USA, 2019. Association for Computing Machinery.

[19] Helia Hashemi, Mohammad Aliannejadi, Hamed Zamani, and W Bruce Croft. Antique: A non-factoid question answering benchmark. In *Proceedings of the 2020 European Conference on Information Retrieval*, ECIR '20, 2020.

[20] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. Improving efficient neural ranking models with cross-architecture knowledge distillation. *CoRR*, abs/2010.02666, 2020.

[21] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. *Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling*, page 113–122. SIGIR '21. Association for Computing Machinery, New York, NY, USA, 2021.

[22] Sebastian Hofstätter, Bhaskar Mitra, Hamed Zamani, Nick Craswell, and Allan Hanbury. Intra-document cascading: Learning to select passages for neural document ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1349–1358, New York, NY, USA, 2021. Association for Computing Machinery.

[23] Sebastian Hofstätter, Hamed Zamani, Bhaskar Mitra, Nick Craswell, and Allan Hanbury. *Local Self-Attention over Long Text for Efficient Document Retrieval*, page 2021–2024. Association for Computing Machinery, New York, NY, USA, 2020.

[24] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[25] Evangelos Kanoulas, Keshi Dai, Virgil Pavlu, and Javed A. Aslam. Score distribution models: Assumptions, intuition, and robustness to score manipulation. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, page 242–249, New York, NY, USA, 2010. Association for Computing Machinery.

[26] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods*

in Natural Language Processing (EMNLP), pages 6769–6781, Online, November 2020. Association for Computational Linguistics.

[27] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Porceedings of the 3rd International Conference on Learning Representations*, ICLR '15, 2015.

[28] Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International Conference on Machine Learning*, pages 3499–3508. PMLR, 2019.

[29] Wouter Kool, Herke van Hoof, and Max Welling. Ancestral gumbel-top-k sampling for sampling without replacement. *J. Mach. Learn. Res.*, 21:47–1, 2020.

[30] Yen-Chieh Lien, Daniel Cohen, and W Bruce Croft. An assumption-free approach to the dynamic truncation of ranked lists. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 79–82, 2019.

[31] Tie-Yan Liu. Learning to rank for information retrieval. 2011.

[32] Craig Macdonald, Rodrygo L. Santos, and Iadh Ounis. The whens and hows of learning to rank for web search. *Inf. Retr.*, 16(5):584–628, oct 2013.

[33] Craig Macdonald and Nicola Tonellotto. Declarative experimentation ininformation retrieval using pyterrier. In *Proceedings of ICTIR 2020*, 2020.

[34] Joel Mackenzie, J. Shane Culpepper, Roi Blanco, Matt Crane, Charles L. A. Clarke, and Jimmy Lin. Query driven algorithm selection in early stage retrieval. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, page 396–404, New York, NY, USA, 2018. Association for Computing Machinery.

[35] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, 2017.

[36] Raghavan Manmatha, Toni Rath, and Fangfang Feng. Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 267–275, 2001.

[37] Ramesh Nallapati. Discriminative models for information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, page 64–71, New York, NY, USA, 2004. Association for Computing Machinery.

[38] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *CoRR*, 2019.

[39] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online, November 2020. Association for Computational Linguistics.

[40] Rama Kumar Pasumarthi, Sebastian Bruch, Xuanhui Wang, Cheng Li, Michael Bendersky, Marc Najork, Jan Pfeifer, Nadav Golbandi, Rohan Anil, and Stephan Wolf. Tf-ranking: Scalable tensorflow library for learning-to-rank. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2970–2978, 2019.

[41] Joaquín Pérez-Iglesias and Lourdes Araujo. Standard deviation as a query hardness estimator. In *Proceedings of the 17th International Conference on String Processing and Information Retrieval*, SPIRE'10, page 207–212, Berlin, Heidelberg, 2010. Springer-Verlag.

[42] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 24(2):193–202, 1975.

[43] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 275–281, New York, NY, USA, 1998. Association for Computing Machinery.

[44] Prafull Prakash, Julian Killingback, and Hamed Zamani. Learning robust dense retrieval models from incomplete relevance labels. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1728–1732, New York, NY, USA, 2021. Association for Computing Machinery.

[45] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847, Online, June 2021. Association for Computational Linguistics.

[46] Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. *CoRR*, abs/2110.07367, 2021.

[47] S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC '96*, pages 109–126, Gaithersburg, Maryland, USA, 1996.

[48] Stephen Robertson, Evangelos Kanoulas, and Emine Yilmaz. Modelling score distributions without actual scores. In *Proceedings of the 2013 Conference on the Theory of Information Retrieval*, pages 85–92, 2013.

[49] Haggai Roitman, Shay Hummel, and Oren Kurland. Using the cross-entropy method to re-rank search results. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, page 839–842, New York, NY, USA, 2014. Association for Computing Machinery.

[50] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, aug 1988.

[51] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. Colbertv2: Effective and efficient retrieval via lightweight late interaction. *CoRR*, abs/2112.01488, 2021.

[52] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. Current challenges and visions in music recommender systems research. *Int. J. Multim. Inf. Retr.*, 7(2):95–116, 2018.

[53] Anna Shtok, Oren Kurland, David Carmel, Fiana Raiber, and Gad Markovits. Predicting query performance by query-drift estimation. *ACM Trans. Inf. Syst.*, 30(2), may 2012.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS '17*, 2017.

[55] Lidan Wang, Jimmy Lin, and Donald Metzler. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 105–114, 2011.

[56] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*, pages 1192–1199, 2008.

[57] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, Jean-Marc Langlois, and Yi Chang. Ranking relevance in yahoo search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 323–332, New York, NY, USA, 2016. Association for Computing Machinery.

[58] Hamed Zamani and W. Bruce Croft. On the theory of weak supervision for information retrieval. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '18, page 147–154, New York, NY, USA, 2018. Association for Computing Machinery.

[59] Hansi Zeng, Hamed Zamani, and Vishwa Vinay. Curriculum learning for dense retrieval distillation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, New York, NY, USA, 2022. Association for Computing Machinery.

[60] Yi Zhang and Jamie Callan. Maximum likelihood estimation for filtering thresholds. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 294–302, New York, NY, USA, 2001. Association for Computing Machinery.