Policy-Gradient Training of Language Models for Ranking

Ge Gao **Jonathan D. Chang** **Claire Cardie** **Kianté Brantley** **Thorsten Joachims**
Department of Computer Science, Cornell University
ggao@cs.cornell.edu {jdc396,ctc9,kdb82}@cornell.edu tj@cs.cornell.edu

Abstract

Text retrieval plays a crucial role in incorporating factual knowledge for decision making into language processing pipelines, ranging from chat-based web search to question answering systems. Current state-of-the-art text retrieval models leverage pre-trained large language models (LLMs) to achieve competitive performance, but training LLM-based retrievers via typical contrastive losses requires intricate heuristics, including selecting hard negatives and using additional supervision as learning signals. This reliance on heuristics stems from the fact that the contrastive loss itself is heuristic and does not directly optimize the downstream metrics of decision quality at the end of the processing pipeline. To address this issue, we introduce Neural PG-RANK, a novel training algorithm that learns to rank by instantiating a LLM as a Plackett-Luce ranking policy. Neural PG-RANK provides a principled method for end-to-end training of retrieval models as part of larger decision systems via policy gradient, with little reliance on complex heuristics, and it effectively unifies the training objective with downstream decision-making quality. We conduct extensive experiments on various text retrieval benchmarks. The results demonstrate that when the training objective aligns with the evaluation setup, Neural PG-RANK yields remarkable in-domain performance improvement, with substantial out-of-domain generalization to some critical datasets employed in downstream question answering tasks.

1 Introduction

Retrieving relevant factual information has become a fundamental component of modern language processing pipelines, as it grounds the decisions of the system and its users in factual sources. In particular, the retrieved text is often utilized by downstream application models to generate accurate outputs for various tasks, ranging from web search (Huang et al., 2013), question answering (Voorhees, 1999; Chen et al., 2017a; Karpukhin et al., 2020), and open-ended generation (Lewis et al., 2020; Paranjape et al., 2022; Yu, 2022). This retrieval process not only acts as a knowledge base and reduces the search space for downstream models, but also can provide users with evidence to understand and validate the model's final output. Consequently, the quality of the retrieval system plays a pivotal role, significantly influencing the accuracy and completeness of any downstream decision making.

Recent research has seen a significant performance boost from incorporating pre-trained large language models into the retrieval policy (e.g., Nogueira & Cho, 2019; Lin et al., 2020; Karpukhin et al., 2020). LLM-based text retrievers excel in contextualizing user queries and documents in natural language, often handling long-form or even conversational inputs. While these neural models generally outperform traditional count-based methods, training high-performing LLM-based retrieval policies presents several challenges.

Foundation Models for Decision Making Workshop at NeurIPS 2023

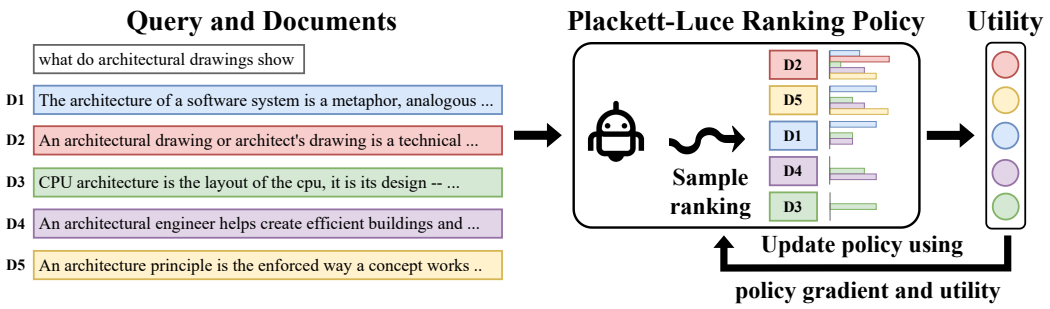

Figure 1: Illustration of our Neural PG-RANK. Given a query and a collection of documents, a Placket-Luce ranking policy samples ranking, receives utility, and gets updated using policy gradient and the received utility. Our method can directly optimize any ranking metric of interest as utility, and allows end-to-end training of any differential policy. Query and document examples are from MS MARCO dataset (Campos et al., 2017).

The primary challenge lies in the complex nature of rankings as combinatorial objects, such that formulating efficient training objectives to enhance LLM-based retrieval functions becomes challenging due to the large number of potential rankings. Existing training methods thus commonly resort to optimizing pairwise preferences as an approximation. Unfortunately, these pairwise training objectives do not directly relate to the desired ranking metrics for retrieval, such as nDCG (Normalised Cumulative Discount Gain) or MRR (Mean Reciprocal Rate). To ameliorate this mismatch, most approaches rely on complex heuristics that are difficult to control, including the careful selection of specific hard negative examples (Xiong et al., 2021), employing a distillation paradigm (Qu et al., 2021; Yang et al., 2020), or adopting an iterative training-and-negative-refreshing approach (Sun et al., 2022). As a result of these intertwined challenges, training a competitive-performing retrieval system is very difficult.

To overcome the above issues, we propose Neural PG-RANK, a rigorous and principled method that directly learns to rank through policy-gradient training. Our approach enables end-to-end training of any differentiable LLM-based retrieval model as a Plackett-Luce ranking policy. Moreover, our method can directly optimize any ranking metric of interest, effectively unifying the training objective with downstream application utility. This enables Neural PG-RANK to not only optimize standard ranking metrics like nDCG, but any application specific metric that evaluates the eventual output of the processing pipeline (e.g., BLEU score). Figure 1 illustrates the proposed Neural PG-RANK framework: given a query and a collection of documents, a Plackett-Luce ranking policy samples rankings, receives utility, and updates itself using policy gradients based on the received utility. By minimizing the need for complex heuristics in negative selection and utilization, as well as eliminating the requirement for additional supervision, our method successfully addresses the aforementioned challenges while establishing a principled bridge between training objectives and downstream utility of retrieval models. Table 1 compares the reliance of state-of-the-art retrieval models, including our Neural PG-RANK, on negative document mining and additional supervision (more details in Section 5).

We conduct extensive experiments employing our Neural PG-RANK with different models on various text retrieval benchmarks. We investigate the effectiveness of our method in both first-stage retrieval (i.e. searching over the entire document collection) and second-stage reranking (i.e. searching within a smaller candidate set per query). The results demonstrate a compelling trend: when the training objective aligns with the evaluation setup, specifically within the context of second-stage reranking, Neural PG-RANK exhibits remarkable in-domain performance improvement. Furthermore, we find substantial out-of-domain generalization from MS MARCO (Campos et al., 2017) to some critical datasets employed in downstream question answering tasks, such as NaturalQuestions (NQ; Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018). Overall, our method and findings pave the way for future research endeavors dedicated to developing highly effective retrieval-based LLM pipelines that are tailored for practical, real-world applications.

Table 1: Reliance of state-of-the-art comparison systems and our Neural PG-RANK on negative document mining and additional supervision. Each check denotes a heuristics used during training. Our method minimizes the reliance on the type of negative documents, and does not require any additional supervision from other models to improve retrieval performance.

Method	**Source of Negative Documents**			**Additional Supervision**	
	In-Batch	BM25	Dense Model	Cross-Encoder	Late Interaction Model
SBERT (Reimers & Gurevych, 2019)		✓	✓✓✓	✓	
TAS-B (Hofstätter et al., 2021)	✓	✓		✓	✓
SPLADEv2 (Formal et al., 2021)		✓	✓✓	✓	
Neural PG-RANK (Ours)			✓		

2 Background and Related Work

Information retrieval (IR) is a class of tasks concerned with searching over a collection to find relevant information to the given query. We focus on text retrieval, where query refers to a user input in natural language and the collection is composed of text documents of arbitrary length. Text retrieval is a central sub-task in many knowledge-intensive NLP problems.

Text Retrieval In the text retrieval literature, retrieval models have evolved from classic count-based methods to recent learning-based neural models. Conventional count-based methods, such as TF-IDF or BM25 (Robertson & Zaragoza, 2009), rely on counting query term occurrences in documents, and do not consider word ordering by treating text as a bag of words. They suffer from issues like lexical mismatch, where relevant documents may not contain exact query terms (Berger et al., 2000). Prior work has explored how to enhance these lexical retrieval methods with neural networks (Nogueira et al., 2019; Cheriton, 2019; Zhao et al., 2021).

Starting from Latent Semantic Analysis (Deerwester et al., 1990), dense vector representations have been studied to improve text retrieval, with recently arising popularity of encoding the query and document as dense vectors (tau Yih et al., 2011; Huang et al., 2013; Gillick et al., 2018). The advent of powerful LLMs has allowed for developing neural models to replace lexical methods, which are often referred as dense models (Nogueira & Cho, 2019; Karpukhin et al., 2020; Humeau et al., 2020). Dense models are typically trained in a supervised manner to differentiate relevant documents from irrelevant ones given the query by assigning higher scores to query-relevant documents. Architectures of commonly-used dense models include bi-encoders (or dual-encoders) which encode query and document separately and compute a similarity score between query and document embeddings (Guo et al., 2020; Liang et al., 2020; Karpukhin et al., 2020; Ma et al., 2021; Ni et al., 2021), cross-encoders which take the concatenation of query and document and output a numerical relevance score (Nogueira & Cho, 2019), and late interaction models which leverage token-level embeddings of query and document from a bi-encoder to compute the final relevance score (Khattab & Zaharia, 2020; Santhanam et al., 2021).

In large-scale text collections, sampling query-irrelevant documents (conventionally called negatives) is necessary for feasible training. Improving negative sampling to obtain a better selection of negatives (i.e. hard negatives) has been an active area of research, such as mining hard negatives from BM25 (Xiong et al., 2021), or from stronger models (Qu et al., 2021; Formal et al., 2021). Another strategy to boost the performance of dense retrieval models is to employ the knowledge distillation paradigm (Qu et al., 2021), where a teacher model can provide query-dependent relevance scores of documents for the student retrieval model to learn from. While negative selection and distillation can improve the retrieval performance, they unfortunately require complex heuristics and convoluted training pipelines. We propose a method that minimizes the reliance on intricate heuristics during training and requires no additional supervision as learning signals. Our method also closes the gap between training objective and evaluation metrics to improve not only the ranking in isolation, but also to directly optimize the overall pipeline performance.

Learning to Rank Learning-to-rank (LTR) has a rich history in the field of IR. Our work falls under the category of LLM-based methods, and for a comprehensive survey of non-LLM-based LTR retrieval models, we refer readers to Liu et al. (2009).

LTR methods used in multi-stage retrieval pipelines have attracted significant interest from both academia (Matveeva et al., 2006; Wang et al., 2011; Asadi & Lin, 2013; Chen et al., 2017b; Mackenzie et al., 2018; Nogueira & Cho, 2019; Khattab & Zaharia, 2020; Luan et al., 2021; Guo et al., 2022) and industry (Delgado & Greyson, 2023). Well-known product deployments of such systems include the Bing search engine (Pedersen, 2010), Alibaba's e-commerce search engine (Liu et al., 2017), and OpenAI's ChatGPT plugins (OpenAI, 2023). The common thread among these studies is the integration of retrieval and ranking systems to ultimately learn effective retrieval strategies.

Among the works in the LTR literature, two that are closely related to our Neural PG-RANK approach are Singh & Joachims (2019) and Oosterhuis (2021), which use Plackett-Luce models to learn a ranking policy. Both approaches extend LTR policies to stochastic policies, allowing for the maximization of task-relevant utility while incorporating fairness constraints during the learning process. In this work, we extend such framework to the context of multi-stage LTR and retrieval pipelines using LLMs, effectively unifying the training objective and ranking evaluation, with additional variance reduction techniques and dense learning signals.

3 Setting

We focus on retrieval problems that involve integrating a text retrieval system into a larger language-processing pipeline. In these applications, user queries can be lengthy and intricate natural language descriptions, and the retrieved results are often used as input for downstream models, which further process them to generate outputs for the overall task. This introduces two requirements that go beyond the traditional retrieval application in search engines. Firstly, the retrieval system must be capable of comprehending complex textual queries, which motivates the utilization of powerful language models as part of the retrieval system. Secondly, it is crucial to optimize the entire set of retrieval results holistically, as the quality of the downstream answer depends on the collective set of retrieval results, rather than individual documents alone.

To address these requirements with a principled machine learning approach, we formalize the problem setting as follows. We assume a distribution $\mathcal{Q}$ from which queries are drawn. Given a query q, we have a candidate set of n documents $\mathbf{d}^q = \{d_1^q, d_2^q, \ldots, d_n^q\}$. Our goal is to train a ranking policy $\pi(r|q)$ that produces a ranking r of the documents in the candidate set $\mathbf{d}^q$ given a query q. For full generality, we allow for stochastic ranking policies, which include deterministic ranking policies as a special case.

To evaluate the quality of a ranking r, we use an application-specific utility function $\Delta(r|q)$. This allows us to define the utility of a ranking policy π for query q as

$$U(\pi|q) = \mathbb{E}_{r\sim\pi(\cdot|q)}\left[\Delta(r|q)\right]. \tag{1}$$

It is worth noting that $\Delta(r|q)$ can be any real-valued and bounded function that measures the quality of the entire ranking r for query q. It does not necessarily need to decompose into relevance judgments of individual documents. For example, $\Delta(r|q)$ can be a function that quantifies the success of using ranking r in a larger language processing pipeline for the overall task, enabling end-to-end optimization of the ranking policy π. Our learning objective is to learn a ranking policy π that optimizes the expected utility over the query distribution $\mathcal{Q}$:

$$\pi^\star = \operatorname*{argmax}_{\pi\in\Pi} \mathbb{E}_{q\sim\mathcal{Q}}\left[U(\pi|q)\right] \tag{2}$$

where Π represents the space of possible ranking policies.

To ensure compatibility with conventional training methods in the retrieval literature, our framework also covers the scenario where we have individual relevance judgments $\mathrm{rel}_i^q \in \{0, 1\}$ for each document in the candidate set, denoted as $\mathbf{rel}^q = \{\mathrm{rel}_1^q, \mathrm{rel}_2^q, \ldots, \mathrm{rel}_n^q\}$. In this case, $\Delta(r|q)$ could be a function like DCG (Cumulative Discount Gain), nDCG (Normalised DCG), MAP (Mean Average Precision), or MRR (Mean Reciprocal Rate). Specifically, for DCG, we have $\Delta_{\text{DCG}}(r|q) = \sum_j \frac{u(r(j)|q)}{\log(1+j)}$ where $u(r(j)|q)$ is the utility of ranking document d_j in the ordering r for the query q. Although our algorithm does not require individual relevance judgments, we focus on the commonly-used nDCG in order to compare with prior that relied on this ranking metric.

4 Method

We present our method, Neural PG-RANK, which addresses the IR problem described in Section 3.

Plackett-Luce Ranking Policy To train our ranking policies, we consider the following functional form that is compatible with any score-based retrieval architecture. In particular, we define representation functions $\eta_\theta^q(q)$ and $\eta_\theta^d(d)$, which encode the query q and the document d into fixed-width vector representations, respectively. Additionally, we introduce a comparison function ϕ which takes these representations and computes a score:

$$s_\theta(q, d) \triangleq \phi(\eta_\theta^q(q), \eta_\theta^d(d))$$

Under the Plackett-Luce model (Plackett, 1975; Luce, 1959), we can define a ranking policy $\pi_\theta(r|q)$ based on the scores $s_\theta(q, d)$. The ranking policy is expressed as a product of softmax distributions:

$$\pi_\theta(r|q) = \prod_{i=1}^{n} \frac{\exp s_\theta(q, d_{r(i)})}{\sum_{j \in \{r(i),\ldots,r(n)\}} \exp s_\theta(q, d_j)}. \tag{3}$$

Note that this family of Plackett-Luce ranking policies includes the policy that simply sorts the documents by their scores as a limiting case:

$$\pi_\theta^{\text{sort}}(r|q) \triangleq \underset{d \in \mathbf{d}^q}{\text{argsort}}\, s_\theta(q, d), \tag{4}$$

where argsort returns the indices that would sort the given array in descending order. In particular, the Plackett-Luce distribution converges to this sort-based policy when the scores are scaled by a factor τ with $\lim \tau \to \infty$. One important distinction between Plackett-Luce policies and sort-based policies is that Plackett-Luce policies remain differentiable, which is a crucial property exploited by our training algorithm. Specifically, our policy $\pi_\theta(r|q)$ and its logarithm $\log \pi_\theta(r|q)$ are differentiable as long as our scoring model s_θ is differentiable.

REINFORCE To solve the optimization problem defined in Equation 2, we propose a policy gradient approach based on insights from the LTR literature (Singh & Joachims, 2019; Oosterhuis, 2021). Using the log-derivative trick pioneered by the REINFORCE algorithm (Williams, 1992), we derive the policy gradient as follows:

$$\begin{aligned} \nabla_\theta U(\pi_\theta|q) &= \nabla_\theta \mathbb{E}_{r \sim \pi_\theta(\cdot|q)} \left[\Delta(r|q)\right] \\ &= \mathbb{E}_{r \sim \pi_\theta(\cdot|q)} \left[\nabla_\theta \log \pi_\theta(r|q) \Delta(r|q)\right]. \end{aligned} \tag{5}$$

Equation 5 exploits the key insight that we can express the gradient of our utility as the expectation over rankings of the gradient of the log-probabilities (i.e. the policy gradient) from our ranking policy π_θ. We can thus estimate Equation 5 using Monte Carlo sampling, as detailed below.

Monte Carlo Sampling A naive method for sampling rankings from the policy π_θ to estimate the gradient is to iteratively draw documents without replacement from the softmax distribution over the remaining documents in the candidate set until there are no more documents left. However, this process has a quadratic computational complexity with respect to the size n of the candidate set. Instead, we can equivalently sample rankings more efficiently in $O(n \log(n))$ time by sampling an entire ranking using the Gumbel-Softmax distribution (Jang et al., 2017) induced by our policy π_θ.

Given a query q and its respective candidate set $\mathbf{d}^q$, to sample an ordering r of documents from our policy π_θ, we first compute the scores $\pi_\theta(r(d)|q)$ for all documents d in the candidate set, as defined in Equation 3. To sample from this induced distribution, we use the Gumbel-Softmax trick. For every document d in the candidate set, we draw independent and identically distributed (i.i.d.) Gumbel samples from the Gumbel distribution $g_d \sim \text{Gumbel}(0, 1)$. Then, we calculate the softmax of the sum of the log scores and their corresponding Gumbel samples as follows:

$$x_d = \frac{\exp(\log \pi_\theta(r(d)|q) + g_d)}{\sum_{d \in \mathbf{d}^q} \exp(\log \pi_\theta(r(d)|q) + g_d)}$$

Finally, we sort the documents according to their x_d values, resulting in the sampled ranking r. In practice, this sampling procedure allows us to sample rankings as fast as we can sort top-K documents, resulting in a $O(n \log(n))$ runtime complexity.

Variance Reduction To reduce the variance induced by our Monte Carlo estimates of the gradient, we incorporate a baseline into our objective. It is important to note that subtracting a baseline from the objective still provides an unbiased estimate of the gradient. Baselines are commonly employed in policy gradient methods to enhance the stability of the updates. In the case of Neural PG-RANK, we adopt the REINFORCE leave-one-out baseline (Kool et al., 2019). The estimation of our policy gradient, based on N Monte Carlo samples, can be expressed as follows:

$$\widehat{\nabla}_\theta U(\pi_\theta|q) = \frac{1}{N}\sum_i \Big[\nabla_\theta \log \pi_\theta(r_i|q)\Big(\Delta(r_i|q) - \frac{1}{N-1}\sum_{j\neq i}\Delta(r_j|q)\Big)\Big]. \tag{6}$$

where r_i is a sampled ranking and q corresponds to a specific query. $\Delta(r_i|q)$ denotes the utility of the ranking r_i for this query q. It subtracts the average utility for all other sampled rankings for this query. By including the leave-one-out baseline, we enhance the estimation of the policy gradient and mitigate the impact of high variance in the updates.

Utility While our Neural PG-RANK applies to any utility function $\Delta(r|q)$, we focus on nDCG@10 in our experiments to be able to compare against conventional methods. Moreover, prior work (e.g., Wang et al., 2013; Thakur et al., 2021) argues that nDCG offers both theoretical consistency and a practical balance suitable for both binary and graded sub-level relevance annotations. Following Oosterhuis (2021), we exploit the insight that the utility at rank k only interacts with the probability of the partial ranking up to k, and the partial ranking after k does not affect the utility before k. The estimation of our policy gradient is now:

$$\begin{aligned}\widehat{\nabla}_\theta U(\pi_\theta|q) = \frac{1}{N}\sum_i \Big[\sum_k \nabla_\theta \log \pi_\theta(r_{i,k}|q, r_{i,1:k-1}) \\ \Big(\text{nDCG}(r_{i,k:}|q, r_{i,1:k-1}) - \frac{1}{N-1}\sum_{j\neq i}\text{nDCG}(r_{j,k:}|q, r_{i,1:k-1})\Big)\Big].\end{aligned} \tag{7}$$

5 Experimental Setup

In numerous applications of text retrieval systems, a prevalent practice involves a two-stage procedure: initially, retrieving a limited set of candidate documents from the full collection (stage 1), and subsequently, re-ranking these initially retrieved candidate documents (stage 2). We investigate the effectiveness of our method in both stages by conducting extensive experiments with different models on various text retrieval benchmarks.

Data We use MS MARCO (Campos et al., 2017), a standard large-scale text retrieval dataset created from real user search queries using Bing search. We train on the train split of MS MARCO from the BEIR benchmark suite (Thakur et al., 2021). For tuning hyperparameters, we carve out a validation set of 7k examples from the training data.

During training, we mimic the two-stage retrieval setup that an eventual production system would use. In particular, we generate candidate sets of 1k documents per query, composed of ground-truth relevant documents to the query and irrelevant documents. These irrelevant documents come from a stage 1 retriever, for which we typically use *gtr-t5-xl* (Ni et al., 2021) model in this work.

For in-domain evaluation, following prior work, we report performance on the dev set of MS MARCO. We also report out-of-domain zero-shot evaluation performance of our MS MARCO models on the subset of BEIR with readily available test sets.[1] BEIR contains several existing text retrieval datasets, ranging from Wikipedia, scientific, financial, and bio-medical domains. Table 5 in Appendix A lists some details of our evaluation sets.

Evaluation Setup We report nDCG@10 (Normalised Cumulative Discount Gain; Järvelin & Kekäläinen, 2000) on each evaluation set by reranking the candidate set per query as a second-stage ranker (Subsection 6.1), or over the full document collection as a first-stage retriever (Subsection 6.2). In the second-stage ranking evaluation, our candidate set for each query comprises of the top-ranked

[1]We include the passage ranking task in TREC-DL 2019 (Craswell et al., 2021), a variant of MS MARCO, as an out-of-domain evaluation set. This dataset is available as the test split of MS MARCO in BEIR.

documents obtained from *gtr-t5-xl* as stage 1 ranker, which serve as irrelevant documents, as well as the ground-truth documents that are known to be relevant to the query. The inclusion of these ground-truth query-relevant documents within the candidate set aims to approximate the candidate set retrieved by an optimal first-stage retriever.

Comparison System We compare to the following systems from prior work:

- **BM25** (Robertson & Zaragoza, 2009) A bag-of-words retrieval approach that ranks a set of documents based on the occurrence of the query tokens in each document using TF-IDF.[2]
- **SBERT** (Reimers & Gurevych, 2019) A bi-encoder, dense retrieval model using hard negatives mined by various systems. The objective combines a negative log likelihood loss and a MarginMSE loss, with reference margin scores generated by a cross-encoder model.[3]
- **TAS-B** (Hofstätter et al., 2021) A bi-encoder model trained with topic-aware queries and a balanced margin sampling technique, replying on dual supervision in a knowledge distillation paradigm. The loss function is a pairwise MarginMSE loss with both hard negatives from BM25 and in-batch negatives.[4]
- **SPLADEv2** (Formal et al., 2021) A bi-encoder model trained by combining a regularization term to learn sparse representation and a MarginMSE loss with hard negatives. Hard negatives and the reference margin scores are generated with a dense model trained with distillation and a cross-encoder reranker.[5]

Excluding BM25, the above supervised learning models are trained on MS MARCO with *distilbert-base-uncased* (Sanh et al., 2019) as the initialization, use dot product to compute query-document similarity, are in comparable scale, and represent the state-of-the-art performance of each approach. Table 1 lists the reliance of these comparison systems and our method on the source of negative documents and additional supervision used during training. Our Neural PG-RANK minimizes the reliance on the type of negative documents, and does not require any additional supervision from other models to improve retrieval performance.

Ranking Policy The representation model η parameterizing our ranking policy is initialized with either SBERT or TAS-B as a warm start.[6] Unless noted in our ablation experiments, we update the policy using our Neural PG-RANK (described in Section 4) for 6 epochs over the training data.

Implementation Detail Our codebase is built upon BEIR (Thakur et al., 2021) and Sentence-Transformers (Reimers & Gurevych, 2019). We run all experiments on A6000 GPUs with 48GB of VRAM. Please see Appendix B for more implementation and hyperparameter details.

6 Experimental Result

For models trained using our method, we present their results on each evaluation set both as a second-stage reranker over the candidate set (Subsection 6.1) and as a first-stage retriever over the full document collection (Subsection 6.2).

6.1 Second-Stage Reranking

We report the performance of our trained models as a second-stage reranker, searching over a candidate set of 1k documents for each query.[7]

[2] `https://github.com/castorini/anserini`

[3] `https://huggingface.co/sentence-transformers/msmarco-distilbert-dot-v5` released on Hugging Face (updated on Jun 15, 2022).

[4] `https://huggingface.co/sentence-transformers/msmarco-distilbert-base-tas-b`

[5] `https://huggingface.co/naver/splade_v2_distil`

[6] Our warmstart models exclude SPLADEv2, because our Neural PG-RANK method does not impose regularization to maintain its sparse representation learning.

[7] BM25 is not compared in second-stage reranking, since it is commonly used only as a first-stage approach.

Table 2: Second-stage reranking: nDCG@10 in-domain results. * marks evaluations run by us using the publicly available checkpoint. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Comparison Systems**			**Ours: Neural PG-RANK**	
	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
MS MARCO dev	0.892	0.893	0.900	**0.987**	0.982

Table 3: Second-stage reranking: nDCG@10 results on out-of-domain datasets. * marks evaluations run by us using the publicly available checkpoint. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Comparison Systems**			**Ours: Neural PG-RANK**	
	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
TREC-DL 2019	0.743	**0.749**	**0.749**	0.742	0.741
TREC-COVID	**0.764**	0.711	0.731	0.690	0.630
NFCorpus	0.308	0.320	**0.341**	0.249	0.303
NQ	0.836	0.836	0.854	0.869	**0.878**
HotpotQA	0.747	0.785	0.834	**0.902**	0.900
FiQA-2018	0.291	0.279	**0.342**	0.131	0.139
ArguAna	0.351	0.479	**0.480**	0.354	0.443
Touché-2020	**0.480**	0.423	0.460	0.363	0.361
Quora	0.962	**0.982**	0.967	0.963	**0.982**
DBPedia	0.513	0.513	**0.533**	0.521	0.525
SCIDOCS	0.144	0.151	**0.163**	0.108	0.136
FEVER	**0.931**	0.911	0.929	0.907	0.913
Climate-FEVER	0.442	0.433	**0.444**	0.438	0.383
SciFact	0.597	0.579	**0.696**	0.316	0.410

In-Domain Performance Table 2 presents the second-stage reranking performance of Neural PG-RANK using various warm-start policies, as measured by nDCG@10. The results reveal that training with Neural PG-RANK leads to remarkable in-domain improvements over the warmstart SBERT and TAS-B models on MS MARCO dev set, with gains of +0.095 and +0.089 in nDCG@10, respectively. Notably, Neural PG-RANK achieves exceptional nDCG scores, approaching a perfect score of 1.0, not only for nDCG@10 (0.987 and 0.982) but also for nDCG@5 (0.986 and 0.981), nDCG@3 (0.985 and 0.978), and nDCG@1 (0.975 and 0.965).[8] In addition, the performance improvements after training with our method are more substantial when measured in nDCG@1, nDCG@3, and nDCG@5. For example, our method yields performance gains of 0.149 and 0.146 over the warm-start SBERT and TAS-B models in nDCG@1. Overall, these in-domain results consistently demonstrate that Neural PG-RANK provides remarkable in-domain performance improvements across various nDCG@k measures, with larger gains observed with smaller k values.

Out-of-Domain Generalization Table 3 shows the second-stage reranking performance of our method on out-of-domain datasets on the BEIR benchmark. In general, models trained with Neural PG-RANK demonstrate a level of generalization comparable to the baseline models. Importantly, they notably outperform the baselines in the case of NaturalQuestions (NQ; Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018), which are critical and widely-studied benchmarks in question answering (QA). Our method achieves strong performance on these datasets, with scores of 0.869/0.878 on NQ and 0.902/0.900 on HotpotQA. Similar to the trend observed in in-domain results across different nNDCG@k measures, our method exhibits larger performance gains with smaller k values in out-of-domain generalization. Remarkably, on HotpotQA, our method using SBERT achieves an impressive nDCG@1 score of 0.974 (see Table 9 in the Appendix). These observations are particularly promising, suggesting that our trained reranker exhibits substantial generalization to the QA domain. We plan to delve deeper into this aspect. Conversely, the datasets in which models trained using our method exhibit comparatively weaker generalization predominantly belong to the domains of science and finance – we hope to investigate this further as well.

[8]We report nDCG@5, nDCG@3 and nDCG@1 of our method for second-stage reranking in Table 7, Table 8 and Table 9 in the Appendix, including both in-domain and out-of-domain evaluation.

Table 4: First-stage retrieval: nDCG@10 in-domain results. * marks evaluations run by us using the publicly available checkpoint. Bold font represents the highest number per row, and underline shows the second highest.

Dataset	**Comparison Systems**				**Ours: Neural PG-RANK**	
	BM25	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
MS MARCO dev	0.228	**0.434**	0.407	<u>0.433</u>	0.416	0.401

Ablation: Training Epochs We investigate how the duration of training impacts the performance of Neural PG-RANK, in both in-domain and out-of-domain scenarios. In Table 10 in the Appendix, we present the results for different training duration, specifically 0, 2, and 6 epochs. These results demonstrate that Neural PG-RANK achieves strong in-domain performance even with just 2 training epochs. However, there is a slight degradation in out-of-domain performance when the training duration is increased to 6 epochs. This suggests that Neural PG-RANK has the potential to enhance its out-of-domain generalization capabilities by carefully selecting the model to strike a balance between in-domain and out-of-domain performance.

6.2 First-Stage Retrieval

In this section, we evaluate Neural PG-RANK in first-stage retrieval, which is to search over the entire document collection for each query. This task can be particularly challenging when dealing with extensive document collections, as is the case when searching through the 8.8 million documents in the MS MARCO dataset.

Table 4 presents the results when we use Neural PG-RANK policies as first-stage retrievers, even though they were trained as a second-stage reranker. We find that training Neural PG-RANK for second-stage reranking is insufficient to match the performance of baseline systems when used as a first-stage retriever.[9] We conjecture that restricting training of Neural PG-RANK to a specific first-stage retriever creates blind-spots in the learned policies, leading to suboptimal performance in first-stage retrieval. To overcome this issue, we will investigate cutting-plane methods, which can enable efficient training even without candidate sets, and which have been shown to be highly effective (and provably convergent) for training other ranking and structured prediction methods (Joachims, 2006; Joachims et al., 2009).

7 Conclusion

In this work, we introduce Neural PG-RANK, a novel training algorithm designed to address challenges associated with training LLM-based retrieval models. As a rigorous approach that reduces the dependence on intricate heuristics and directly optimizes relevant ranking metrics, Neural PG-RANK has demonstrated its effectiveness when training objective aligns with evaluation setup — specifically, in the context of second-stage reranking — by exhibiting remarkable in-domain performance improvement and presenting substantial out-of-domain generalization to some critical datasets employed in downstream question answering. Our work establishes a principled bridge between training objectives and practical utility of the collective set of retrieved results, thereby paving the way for future research endeavors aimed at constructing highly effective retrieval-based LLM pipelines that are tailored for practical applications.

Acknowledgments

This research was supported in part by NSFAwards IIS-1901168, IIS-2312865 and OAC-2311521. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors. We thank Daniel D. Lee, Travers Rhodes, Chanwoo Chun, and Minh Nguyen for helpful discussions and support.

[9] We observe the same finding in the out-of-domain evaluation, which is reported in Table 11 in the Appendix.

References

Nima Asadi and Jimmy Lin. Effectiveness/efficiency tradeoffs for candidate generation in multi-stage retrieval architectures. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2013.

Adam L. Berger, Rich Caruana, David A. Cohn, Dayne Freitag, and Vibhu Mittal. Bridging the lexical chasm: statistical approaches to answer-finding. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2000.

Daniel Fernando Campos, Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, Li Deng, and Bhaskar Mitra. Ms marco: A human generated machine reading comprehension dataset. *International Conference on Learning Representations*, 2017.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *Annual Meeting of the Association for Computational Linguistics*, 2017a.

Ruey-Cheng Chen, Luke Gallagher, Roi Blanco, and J Shane Culpepper. Efficient cost-aware cascade ranking in multi-stage retrieval. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017b.

David R. Cheriton. From doc2query to doctttttquery. *ArXiv*, 2019.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Fernando Campos, and Ellen M. Voorhees. Overview of the trec 2020 deep learning track. *ArXiv*, 2021.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the Association for Information Science and Technology*, 1990.

Joaquin Delgado and Paul Greyson. From structured search to learning to rank and retrieve. Blog, March 2023. URL `https://www.amazon.science/blog/from-structured-search-to-learning-to-rank-and-retrieve`. Accessed: June 23, 2023.

Thibault Formal, C. Lassance, Benjamin Piwowarski, and Stéphane Clinchant. Splade v2: Sparse lexical and expansion model for information retrieval. *ArXiv*, 2021.

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. End-to-end retrieval in continuous space. *ArXiv*, 2018.

Jiafeng Guo, Yinqiong Cai, Yixing Fan, Fei Sun, Ruqing Zhang, and Xueqi Cheng. Semantic models for the first-stage retrieval: A comprehensive review. *ACM Transactions on Information Systems*, 2022.

Mandy Guo, Yinfei Yang, Daniel Matthew Cer, Qinlan Shen, and Noah Constant. Multireqa: A cross-domain evaluation forretrieval question answering models. *European Chapter of the Association for Computational Linguistics: The Second Workshop on Domain Adaptation for NLP*, 2020.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy J. Lin, and Allan Hanbury. Efficiently teaching an effective dense retriever with balanced topic aware sampling. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. *ACM International Conference on Information & Knowledge Management*, 2013.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. *International Conference on Learning Representations*, 2020.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *International Conference on Learning Representations*, 2017.

Kalervo Järvelin and Jaana Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. *International ACM SIGIR Conference on Research and Development in Information Retrieval: Forum*, 2000.

T. Joachims. Training linear SVMs in linear time. *ACM SIGKDD International Conference On Knowledge Discovery and Data Mining*, 2006.

T. Joachims, T. Finley, and Chun-Nam Yu. Cutting-plane training of structural svms. *Machine Learning*, 2009.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Yu Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. Dense passage retrieval for open-domain question answering. *Conference on Empirical Methods in Natural Language Processing*, 2020.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. *International ACM SIGIR conference on research and development in Information Retrieval*, 2020.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! *International Conference on Learning Representations: Deep RL Meets Structured Prediction Workshop*, 2019.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 2019.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Conference on Neural Information Processing Systems*, 2020.

Davis Liang, Peng Xu, Siamak Shakeri, Cícero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. Embedding-based zero-shot retrieval through query generation. *ArXiv*, 2020.

Jimmy J. Lin, Rodrigo Nogueira, and Andrew Yates. Pretrained transformers for text ranking: Bert and beyond. *ACM International Conference on Web Search and Data Mining*, 2020.

Shichen Liu, Fei Xiao, Wenwu Ou, and Luo Si. Cascade ranking for operational e-commerce search. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 2009.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 2021.

R Duncan Luce. *Individual choice behavior: A theoretical analysis*. 1959.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith B. Hall, and Ryan T. McDonald. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. *Conference of the European Chapter of the Association for Computational Linguistics*, 2021.

Joel Mackenzie, J Shane Culpepper, Roi Blanco, Matt Crane, Charles LA Clarke, and Jimmy Lin. Query driven algorithm selection in early stage retrieval. *ACM International Conference on Web Search and Data Mining*, 2018.

Irina Matveeva, Chris Burges, Timo Burkard, Andy Laucius, and Leon Wong. High accuracy retrieval with multiple nested ranker. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith B. Hall, Ming-Wei Chang, and Yinfei Yang. Large dual encoders are generalizable retrievers. *Conference on Empirical Methods in Natural Language Processing*, 2021.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *ArXiv*, 2019.

Rodrigo Nogueira, Wei Yang, Jimmy J. Lin, and Kyunghyun Cho. Document expansion by query prediction. *ArXiv*, 2019.

Harrie Oosterhuis. Computationally efficient optimization of plackett-luce ranking models for relevance and fairness. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

OpenAI. Chatgpt plugins: Extending conversational ai. Blog, March 2023. URL `https://openai.com/blog/chatgpt-plugins`. Accessed: June 23, 2023.

Ashwin Paranjape, O. Khattab, Christopher Potts, Matei A. Zaharia, and Christopher D. Manning. Hindsight: Posterior-guided training of retrievers for improved open-ended generation. *International Conference on Learning Representations*, 2022.

Jan Pedersen. Query understanding at bing, 2010.

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 1975.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. *North American Chapter of the Association for Computational Linguistics*, 2021.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *Conference on Empirical Methods in Natural Language Processing*, 2019.

Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 2009.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *Conference on Neural Information Processing Systems: The Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing*, 2019.

Keshav Santhanam, O. Khattab, Jon Saad-Falcon, Christopher Potts, and Matei A. Zaharia. Colbertv2: Effective and efficient retrieval via lightweight late interaction. *North American Chapter of the Association for Computational Linguistics*, 2021.

Ashudeep Singh and Thorsten Joachims. Policy learning for fairness in ranking. *Advances in neural information processing systems*, 2019.

Si Sun, Chenyan Xiong, Yue Yu, Arnold Overwijk, Zhiyuan Liu, and Jie Bao. Reduce catastrophic forgetting of dense retrieval training with teleportation negatives. *Conference on Empirical Methods in Natural Language Processing*, 2022.

Wen tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. Learning discriminative projections for text similarity measures. *Conference on Computational Natural Language Learning*, 2011.

Nandan Thakur, Nils Reimers, Andreas Ruckl'e, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *Conference on Neural Information Processing Systems*, 2021.

Ellen M. Voorhees. The trec-8 question answering track report. *Text Retrieval Conference*, 1999.

Lidan Wang, Jimmy Lin, and Donald Metzler. A cascade ranking model for efficient ranked retrieval. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2011.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. A theoretical analysis of ndcg type ranking measures. *Annual Conference Computational Learning Theory*, 2013.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*, 1992.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *International Conference on Learning Representations*, 2021.

Yinfei Yang, Ning Jin, Kuo Lin, Mandy Guo, and Daniel Matthew Cer. Neural retrieval for question answering with cross-attention supervised data augmentation. *Annual Meeting of the Association for Computational Linguistics*, 2020.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *Conference on Empirical Methods in Natural Language Processing*, 2018.

W. Yu. Retrieval-augmented generation across heterogeneous knowledge. *North American Chapter of the Association for Computational Linguistics*, 2022.

Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. Sparta: Efficient open-domain question answering via sparse transformer matching retrieval. *North American Chapter of the Association for Computational Linguistics*, 2021.

Table 5: Details of our evaluation sets (test set unless noted otherwise): source domain of documents (Domain), number of queries (# Q), number of documents in the full collection, (# D), average number of relevant documents per query (# Rel. D/Q), and the type of relevance annotation (Annotation).

Dataset	**Domain**	**# Q**	**# D**	**# Rel. D/Q**	**Annotation**
MS MARCO dev	misc.	6,980	8.8M	1.1	binary
TREC-DL 2019	misc.	43	9.1k	95.4	3-level
TREC-COVID	bio-medical	50	171.3k	439.5	3-level
NFCorpus	bio-medical	323	3.6k	38.2	3-level
NQ	Wikipedia	3,452	2.7M	1.2	binary
HotpotQA	Wikipedia	7.405	5.2M	2.0	binary
FiQA-2018	finance	648	57.6k	2.6	binary
ArguAna	misc.	1,406	8.7k	1.0	binary
Touché-2020	misc.	49	382.5k	19.0	3-level
Quora	Quora	10,000	522.9k	1.6	binary
DBPedia	Wikipedia	400	4.6M	38.2	3-level
SCIDOCS	scientific	1,000	25.7k	4.9	binary
FEVER	Wikipedia	6,666	5.4M	1.2	binary
Climate-FEVER	Wikipedia	1,535	5.4M	3.0	binary
SciFact	scientific	300	5,2k	1.1	binary

Table 6: Hyperparameters used for Neural PG-RANK.

Setting	**Values**
model	[SBERT, TAS-B]
Neural PG-RANK	epochs: 6 batch size: 220 learning rate: 1e-6 entropy coeff: 0.01 # rankings sampled per epoch: 5000 gumbel softmax temperature (τ): 0.05 similarity function: dot product

A Dataset Statistics

Table 5 reports some details of the evaluation datasets in BEIR that we report performance on. Most evaluation sets have binary annotation of the document relevance given the query (i.e. either relevant or irrelevant to the query), while some datasets provide graded annotation of the document relevance into sub-levels – a grade of 0 means irrelevant, and positive grades (e.g., 3-level annotation gives 1, 2, or 3 as relevance judgement) marks relevant document.

B Implementation Detail

Table 6 lists the hyperparameters used in our experiments. Note that we use the same training hyperparameters across all experiments with different warmstart models in our work.

C Performance Tables

Second-Stage Reranking In addition to NDCG@10 reported in Subsection 6.1, we report NDCG@1 in Table 9, NDCG@3 in Table 8, and NDCG@5 in Table 7 for the second-stage reranking performance of our models trained with Neural PG-RANK. Table 10 shows the performance at 0, 2, and 6 epochs of training. 0 epoch means the warmstart models.

First-Stage Retrieval Table 11 reports evaluation of our models trained on MS MARCO as a first-stage retriever on out-of-domain datasets in BEIR.

Table 7: Second-stage reranking: nDCG@5 results. * marks evaluations run by us using the publicly available checkpoint. ‡ double dagger symbol means in-domain evaluation. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Comparison Systems**			**Ours: Neural PG-RANK**	
	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
MS MARCO dev‡	0.884	0.884	0.892	**0.986**	0.981
TREC-DL 2019	0.753	0.765	0.757	0.767	**0.771**
TREC-COVID	**0.782**	0.719	0.758	0.717	0.659
NFCorpus	0.338	0.356	**0.376**	0.281	0.334
NQ	0.822	0.822	0.842	0.860	**0.868**
HotpotQA	0.730	0.769	0.819	**0.892**	0.890
FiQA-2018	0.267	0.251	**0.317**	0.122	0.127
ArguAna	0.291	**0.435**	0.426	0.307	0.395
Touché-2020	**0.526**	0.439	0.516	0.382	0.378
Quora	0.959	**0.981**	0.964	0.960	**0.981**
DBPedia	0.517	0.513	**0.529**	0.524	0.514
SCIDOCS	0.122	0.127	**0.134**	0.092	0.114
FEVER	**0.925**	0.904	0.923	0.902	0.908
Climate-FEVER	0.371	0.388	0.398	**0.404**	0.350
SciFact	0.575	0.558	**0.674**	0.279	0.379

Table 8: Second-stage reranking: nDCG@3 results. * marks evaluations run by us using the publicly available checkpoint. ‡ double dagger symbol means in-domain evaluation. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Comparison Systems**			**Ours: Neural PG-RANK**	
	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
MS MARCO dev‡	0.872	0.872	0.881	**0.985**	0.978
TREC-DL 2019	0.748	0.764	0.758	**0.772**	0.770
TREC-COVID	**0.810**	0.745	0.770	0.735	0.669
NFCorpus	0.364	0.385	**0.405**	0.305	0.364
NQ	0.804	0.806	0.821	0.846	**0.857**
HotpotQA	0.712	0.749	0.799	**0.878**	0.875
FiQA-2018	0.260	0.244	**0.302**	0.123	0.124
ArguAna	0.245	**0.385**	0.368	0.268	0.349
Touché-2020	**0.549**	0.467	0.540	0.404	0.418
Quora	0.955	0.979	0.960	0.956	**0.979**
DBPedia	**0.539**	0.526	0.533	**0.539**	0.528
SCIDOCS	0.140	0.151	**0.152**	0.108	0.133
FEVER	**0.921**	0.898	0.918	0.895	0.902
Climate-FEVER	0.350	0.369	0.379	**0.401**	0.346
SciFact	0.563	0.534	**0.662**	0.260	0.353

Table 9: Second-stage reranking: nDCG@1 results. * marks evaluations run by us using the publicly available checkpoint. ‡ double dagger symbol means in-domain evaluation. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Comparison Systems**			**Ours: Neural PG-RANK**	
	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
MS MARCO dev‡	0.826	0.819	0.830	**0.975**	0.965
TREC-DL 2019	0.771	0.764	0.795	**0.802**	0.744
TREC-COVID	**0.810**	0.740	0.770	0.770	0.700
NFCorpus	0.406	0.438	**0.460**	0.344	0.410
NQ	0.758	0.752	0.770	0.815	**0.822**
HotpotQA	0.884	0.904	0.941	**0.974**	**0.974**
FiQA-2018	0.286	0.265	**0.329**	0.150	0.140
ArguAna	0.147	**0.245**	0.237	0.171	0.233
Touché-2020	**0.561**	0.510	**0.561**	0.449	0.439
Quora	0.946	0.975	0.952	0.950	**0.976**
DBPedia	**0.618**	0.570	0.585	0.604	0.583
SCIDOCS	0.182	0.187	**0.196**	0.142	0.176
FEVER	**0.928**	0.889	0.916	0.885	0.893
Climate-FEVER	0.432	0.446	0.453	**0.536**	0.463
SciFact	0.473	0.470	**0.603**	0.217	0.283

Table 10: Second-stage reranking: nDCG@10 results of evaluating the warmstart model, the model after training for 2 epochs and after 6 epochs. ‡ double dagger symbol means in-domain evaluation. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Performance of Neural PG-RANK at Epoch 0 → 2 → 6**	
	with SBERT	with TAS-B
MS MARCO dev‡	0.892 → 0.982 → 0.987	0.893 → 0.963 → 0.982
Avg. on other BEIR datasets	0.579 → 0.546 → 0.539	0.582 → 0.573 → 0.553

Table 11: First-stage retrieval: nDCG@10 results on out-of-domain datasets. * marks evaluations run by us using the publicly available checkpoint. Bold font represents the highest number per row, and underline shows the second highest. Light green color highlights the experiments where our Neural PG-RANK yields performance gain.

Dataset	**Comparison Systems**				**Ours: Neural PG-RANK**	
	BM25	SBERT*	TAS-B*	SPLADEv2*	with SBERT	with TAS-B
TREC-DL 2019	0.506	0.703	0.723	**0.729**	0.703	0.710
TREC-COVID	0.656	0.664	0.487	**0.710**	0.446	0.346
NFCorpus	0.325	0.298	0.315	**0.334**	0.147	0.243
NQ	0.329	0.498	0.455	**0.521**	0.384	0.386
HotpotQA	0.603	0.587	0.581	**0.684**	0.500	0.465
FiQA-2018	0.236	0.286	0.276	**0.336**	0.124	0.133
ArguAna	0.315	0.349	**0.479**	**0.479**	0.353	0.442
Touché-2020	**0.367**	0.224	0.171	0.272	0.129	0.110
Quora	0.789	0.833	0.835	0.838	**0.839**	0.832
DBPedia	0.313	0.375	0.385	**0.435**	0.365	0.358
SCIDOCS	**0.158**	0.141	0.145	**0.158**	0.085	0.096
FEVER	0.753	0.774	0.678	**0.786**	0.358	0.341
Climate-FEVER	0.213	**0.235**	0.193	**0.235**	0.044	0.035
SciFact	0.665	0.595	0.575	**0.693**	0.264	0.369