Beyond Monolithic Finetuning: Evaluating Query-Length-Specific Finetuning and Ensembling for Neural Ranking Models

Serkan Akin, Elvira Antonogiannaki, Yiming Chen, Yulin Chen
Group 19

Abstract

Neural ranker performance often varies with query characteristics like length. We investigate fine-tuning lightweight models specialized for short, medium, or long queries, combined with dynamic length-based routing at inference. Evaluating on standard IR benchmarks (nDCG, and Recall) against a monolithic baseline, we test both routed specialized models and simple ensembles. This query-length-aware approach aims to improve retrieval effectiveness and potentially efficiency, offering a strategy for adaptive neural ranking.

CCS Concepts

• **Information systems → Retrieval models and ranking**.

Keywords

Neural Ranking Models, Information Retrieval, Query Length, Fine-Tuning, Dynamic Routing, Ensemble Methods

1 Introduction

1.1 Background and Motivation

In recent years, neural ranking models have significantly advanced the state of the art in information retrieval (IR), particularly in tasks such as document ranking, passage retrieval, and question answering [14, 15]. Transformer-based architectures—such as BERT and its variants—are central to modern retrieval pipelines due to their capacity to capture rich contextual representations [7, 12]. Typically, these models operate within a two-stage retrieval framework: an initial sparse or dense retriever selects candidate documents, which are then re-ranked using a neural re-ranker.

However, the application of a single, monolithic ranking model across all query types presupposes uniform performance across diverse query characteristics. In practice, user queries differ greatly in length and complexity. Short queries, often keyword-based or navigational, are syntactically sparse and semantically ambiguous. Longer queries, like natural language questions, typically offer more context and clearer expressions of information needs. Prior studies highlight that query length significantly influences retrieval effectiveness, generally associating longer queries with improved precision and recall due to richer semantic context [3]. Nevertheless, most existing neural ranking frameworks overlook query length as a meaningful signal during both training and inference phases.

Additionally, deploying large neural models in real-time retrieval systems incurs substantial computational costs. High-performance search and QA applications must balance retrieval effectiveness with efficiency constraints. Lightweight transformer models, such as MiniLM, DistilBERT, and DistilRoBERTa, aim to address this challenge by retaining substantial accuracy while significantly reducing inference latency and memory usage [17]. Yet, these compact models typically serve as general-purpose solutions without query-specific optimization.

This study investigates the hypothesis that retrieval effectiveness and efficiency can be improved by explicitly incorporating query length into both model training and inference processes. We explore whether specialized lightweight models fine-tuned on short, medium, and long queries, combined with dynamic inference-time routing, outperform a single, general-purpose model.

1.2 Problem Statement

Most contemporary IR systems utilize a single ranking model fine-tuned without considering variations in query length, resulting in several limitations:

- **Lack of Specialization:** A single model may inadequately generalize across different query lengths, potentially underfitting long queries or overfitting short ones.
- **Computational Inefficiency:** Uniformly applying a computationally intensive model to all queries, particularly simple short queries, leads to unnecessary latency and resource consumption.
- **Static Inference Approach:** Traditional retrieval pipelines do not adapt inference dynamically based on observable query features such as token length.

To address these limitations, we introduce a neural ranking approach centered on query length. This involves fine-tuning specialized lightweight transformer models for distinct query-length categories (short, medium, long). At inference, queries are either dynamically routed to the corresponding specialized model or predictions from these models are ensembled. This strategy aims to enhance retrieval effectiveness while offering potential gains in computational efficiency.

1.3 Goals and Contributions

In this paper, we examine the retrival effectiveness of query-length-specific fine-tuning and ensembling strategies in neural ranking for IR tasks. Our primary contributions include:

- Proposing a dynamic neural ranking pipeline that classifies queries into short, medium, or long based on token length

DSAIT4050-Q3-25, TU Delft

https://doi.org/10.1145/nnnnnnn.nnnnnnn

and either routes them to corresponding lightweight transformer models or combines predictions from these models, all fine-tuned on length-specific training subsets.
- Empirically validating whether query-length-specific fine-tuning significantly improves retrieval effectiveness compared to monolithic fine-tuning on the entire dataset.
- Investigating various ensembling methods — such as simple averaging, fixed weighting, and learned weighting — to determine if combining specialized models further enhances retrieval performance.
- Evaluating our proposed approaches on two publicly available datasets, FiQA and Quora, demonstrating their impact on retrieval effectiveness using standard IR metrics.

Specifically, we aim to answer the following research questions:

- **RQ1:** What is the impact of query-length-specific fine-tuning on retrieval performance compared to monolithic fine-tuning?
- **RQ2:** How do different ensembling strategies influence retrieval performance when combining query-length-specific models?

2 Related Work

2.1 Query Length and Neural Ranking

While neural retrieval models like DPR [11] and ColBERT [12] often outperform sparse methods (e.g., BM25), benchmarks like BEIR show varying performance, indicating no single approach is universally best [18]. **Query length** is a critical factor; shorter queries may suit keyword matching, while longer ones benefit from contextual neural models [5, 16]. However, even neural models can struggle with very long or short queries [16]. Some works address this via query expansion or model adaptations [12], but often still rely on a single model.

2.2 Ensembling and Dynamic Routing in IR

Ensemble methods combine multiple rankers to improve robustness. Simple approaches like score averaging or Reciprocal Rank Fusion (RRF) merge outputs with low overhead [4]. More complex methods use learned weights [8, 13] or adapt weights based on query features like length [2].

Building on this, **dynamic model selection** or routing chooses the best model(s) at inference time based on query characteristics [6]. Modern techniques employ query performance prediction (QPP) or mixture-of-experts (MoE) architectures [8, 10] to route queries, often using lightweight models (e.g., TinyBERT) as "fast experts" for simpler queries [20]. This selective routing can enhance both accuracy and efficiency [8]. Our work aligns with this dynamic paradigm but focuses specifically on specializing multiple *lightweight* models based *purely on query length tertiles* and evaluating simple routing (selection) alongside basic ensembling strategies.

3 Methodology

The query-length-aware approach proposed consists of three query-length-specific fine-tuned models (i.e. short, medium, and long) that are then ensembled at inference time to produce a final relevance score for each query-document pair. The goal is to assess whether employing query-length-specific fine-tuning and ensembling offers improved performance over standard monolithic fine-tuning for information retrieval.

3.1 Datasets

We evaluate our approach on two publicly available IR datasets: **FiQA** (financial QA) and **Quora Question Pairs** (general paraphrase QA). Both originate from the BEIR repository [18], but we opted not to include other BEIR sets due to data scale, domain overlap, and the limited availability of large-scale relevance judgments in some cases.

Why FiQA and Quora from BEIR?. Although BEIR offers numerous datasets (e.g., SciFact, TREC-COVID, NFCorpus, ArguAna), we focus on FiQA and Quora for three main reasons:

(1) **Sufficient Labeled Data:** Some BEIR collections lack large-scale, high-quality relevance labels or emphasize tasks beyond passage-level retrieval. By contrast, FiQA and Quora each have well-defined query-document or query-question pairs, enabling standard IR evaluation.
(2) **Domain Coverage and Query Diversity:** FiQA is finance-oriented, containing moderately lengthy queries with domain-specific terms, while Quora covers broad user-generated questions that can be extremely short or very long. This contrast naturally supports our query-length-based approach.
(3) **Computational Constraints:** Training multiple specialized models on more than two large datasets would exceed our resource budget. FiQA and Quora offer a balanced scale, capturing a wide range of query lengths without excessive compute requirements.

FiQA.. FiQA is a finance QA dataset containing user queries about stocks, personal finance, and investment topics. Table 1 shows that FiQA's test queries average 10.94 tokens (std 4.30), ranging from 2 to 31 tokens. Notably, around one-third of queries fall below 9 tokens, while another third exceed 13 tokens. On average, each FiQA query has about 2.63 relevant documents, but some queries can link to as many as 15, indicating that many finance questions have multiple valid answers.

Quora Question Pairs. Quora is a paraphrase-focused dataset featuring user-generated questions. Its test queries average 9.53 tokens (std 4.06), ranging from 1 to 43 (Table 1). Many queries are concise (2–5 tokens), while others are substantially longer. In terms of relevance, Quora queries typically have 1.63 relevant matches each, with the majority having exactly one paraphrase, although a few outliers have up to 34 matches.

Table 1: Key Statistics for FiQA and Quora Test Sets

Dataset	Count	Mean Len	Min–Max	Rel (avg/max)
FiQA	648	10.94	2–31	2.63 / 15
Quora	650[1]	9.53	1–43	1.63 / 34

[1] The Quora dataset from BEIR does not have separate ***train*** and ***test*** sets. Therefore, we split the test set (10,000 queries) into ***train***, ***val***, and ***test***.

Overall, the specialized (FiQA) and general (Quora) domains, paired with diverse query lengths and varying relevance counts, make these datasets an ideal testbed for evaluating our query-length-aware finetuning and dynamic routing approach.

3.2 Query Length Bucketing

To define the length thresholds for short, medium, and long queries, we divide each dataset into three equally sized buckets using the 33rd and 67th percentiles of query length (measured in words). This data-driven approach adapts to each dataset's unique distribution, and characteristics, avoiding arbitrary cutoffs that may not generalize well. For instance, while about a third of queries in both Quora and FiQA have less than 8 words, their medium-length ranges differ: Quora spans 9-10 words, FiQA 9-12. If we expanded Quora's medium bucket to include longer queries, the more frequent 9–10 word queries would dominate the training data, possibly making the model less effective at handling longer, less common queries. Tertile-based bucketing ensures each model is trained on a balanced, representative subset of the data.

3.3 Training

3.3.1 Models Under Comparison. In order to evaluate the performance of the system, three lightweight Transformer models of varying sizes are fine-tuned: MiniLM-L12-H384-uncased (33M) [19] and DistilBERT (66M) [17], both distilled versions of BERT, as well as DistilRoBERTa (82M) [17], a distilled version of RoBERTa, which is an adaptation of the BERT model with improved pertaining techniques. All three models are known to offer considerably lower memory consumption and faster inference times, while maintaining strong accuracy compared to the original models [17, 19]. This characteristic allows us to evaluate the proposed system in settings that may resemble real-world applications, including resource-constrained environments and retrieval pipelines where inference speed is critical. Furthermore, each model adopts a distinct distillation strategy, or base model, resulting in different trade-offs in terms of model size, inference speed, and accuracy. Therefore, we are able to investigate the effectiveness of the query-length-aware routing strategy in terms of its generalizability across multiple compact Transformer architectures, and sizes.

3.3.2 Model Fine-Tuning. The fine-tuning approach followed is inspired by the Dense Passage Retrieval (DPR) [11], in which bi-encoder models are trained using independently encoded queries and documents, and optimized with a contrastive loss. In our case, each query and document is independently tokenized using the tokenizer of the corresponding pre-trained model and passed through its encoder to obtain a fixed-size embedding. Each model is fine-tuned using a margin-based triplet loss with dot product similarity to encourage a higher similarity between the relevant pairs. We also employed a batch-wise negative sampling strategy, where negatives are selected from other query-document pairs within the same batch while ensuring that known positives are excluded. This promotes efficient training while reducing the risk of false negatives. We fine-tune a separate model for each combination of transformer architecture and query-length category (short, medium, and long), with training performed independently in all cases. By fine-tuning separately for each query-length bucket, we aim to enable each model to specialize in the semantic and structural patterns typical of different query types.

3.4 Dynamic Routing and Ensembling at Inference Time

The three query-length-specific fine-tuned models are ensembled at inference time, using four different strategies, selected to explore the trade-off between model specialization, effectiveness, and computational efficiency. **Dynamic Selection (Routing)** selects and runs only the model trained on the query's length category, leveraging full specialization of the fine-tuned model and offering the most efficient inference strategy. **Ensemble Average** takes a more simple approach, where the final relevance score is computed as the uniform average of the predictions from all three models. **Ensemble Weighted Fixed** naively assigns a higher weight (0.5) to the model trained on the query's corresponding length category and lower weights (0.25) to the others to reflect a heuristic assumption that specialization is beneficial. Finally, **Ensemble Learned Weighted** uses ***Optuna*** [1] to sweep over different weight configurations, with average nDCG@100 on validation data as the benchmark. The configuration that scored the highest is then used for evaluation. While the last three strategies may benefit from combining multiple perspectives, they require executing all models at inference time, making them more computationally expensive. In contrast, dynamic routing achieves better computation efficiency by passing the query through one model at a time instead of three, which may come at the cost of reduced effectiveness as the strict, length-based selection sacrifices the robustness gained from combining multiple perspectives.

All strategies are evaluated against a monolithic baseline for each transformer architecture (Section 3.3.1), in which each model is fine-tuned on the entire dataset without query-length-specific segmentation. The Recall (R@100), and Normalized Discounted Cumulative Gain (nDCG@100) on the top-100 documents are employed. R@100 evaluates the model's ability to retrieve all relevant documents, which is critical in first-stage retrieval as it ensures that the downstream components, such as re-rankers have access to complete candidate sets [9]. nDCG@100 evaluates whether the most relevant documents tend to appear near the top on the retrieval list, which is particularly useful in cases where re-ranking is omitted. This combined evaluation provides insights into both the coverage and ranking robustness of the models and ensemble strategies.

4 Results

4.1 Overall Performance Comparison

Tables 2 and 3 present the main results for FiQA and Quora, respectively, comparing the monolithic baseline against our length-aware approaches across the three base model architectures. Bold values indicate the best performance for each metric within a specific base model column.

4.2 Discussion

RQ:1 Impact of Query-Length-Specific Fine-Tuning (vs. Monolithic). To assess the direct impact of specialization without ensembling,

Table 2: Performance on FiQA test set (p-value for nDCG)

Method	MiniLM			DistilBERT			DistilRoBERTa		
	nDCG@100	R@100	p-value	nDCG@100	R@100	p-value	nDCG@100	R@100	p-value
Baseline (Monolithic)	**0.1749**	**0.3961**	-	0.2213	0.4707	-	0.2104	0.4409	-
Selection (Routing)	0.1250	0.3075	<0.0001	0.1964	0.4159	0.0019	0.1773	0.3872	0.0008
Avg Ensemble	0.1304	0.3312	<0.0001	0.2247	0.4559	0.6505	0.2206	0.4686	0.2755
Fixed W Ensemble	0.1606	0.3711	0.0859	**0.2347**	**0.4712**	0.0724	0.2390	0.4969	0.0022
Learned W Ensemble	0.1572	0.3657	0.0313	0.2296	0.4599	0.2858	**0.2448**	**0.5069**	0.0003

Table 3: Performance on Quora test set (p-value for nDCG)

Method	MiniLM			DistilBERT			DistilRoBERTa		
	nDCG@100	R@100	p-value	nDCG@100	R@100	p-value	nDCG@100	R@100	p-value
Baseline (Monolithic)	0.7090	0.8931	-	0.7617	0.9559	-	0.6302	0.8560	-
Selection (Routing)	0.7177	0.8915	<0.0001	0.7348	0.9287	0.0180	0.6357	0.8741	0.7256
Avg Ensemble	0.6553	0.8858	0.5251	0.7566	0.9575	0.6373	0.6702	0.9238	0.0026
Fixed W Ensemble	**0.7665**	**0.9354**	<0.0001	0.7988	0.9688	0.0003	**0.7529**	**0.9523**	<0.0001
Learned W Ensemble	0.7598	0.9350	<0.0001	**0.8053**	**0.9713**	<0.0001	0.7518	0.9465	<0.0001

we compare the 'Selection' (dynamic routing) strategy against the 'Baseline' (monolithic).

On **FiQA** (Table 2), dynamic selection consistently and significantly *underperformed* the monolithic baseline across all three models for both nDCG@100 and Recall@100. For instance, with DistilBERT, Selection nDCG@100 (0.1964) was substantially lower than the baseline's 0.2213.

On **Quora** (Table 3), the results were mixed: Selection slightly improved nDCG@100 for MiniLM (0.7177 vs 0.7090) and both metrics for DistilRoBERTa (nDCG@100: 0.6357 vs 0.6302; R@100: 0.8741 vs 0.8560), but decreased performance for DistilBERT.

Conclusion for RQ1: Simply routing to a specialized model ('Selection') does not reliably outperform a monolithic baseline, despite having the same single-model inference cost. Significant degradation occurred on **FiQA**, and gains on **Quora** were inconsistent across models. This suggests specialization alone based solely on query length is not an effective strategy, likely because length is an inadequate proxy for the complexities dictating optimal model handling. Additionally, the specialized model is only trained on a subset of the data, which can adversely affect the model's ability to generalize, especially on smaller/specialized datasets.

RQ2: Effectiveness of Ensembling Strategies. To evaluate how different ensembling methods influence performance, we compare Average Ensemble ('Avg Ens'), Fixed Weighted Ensemble ('Fixed W Ens'), and Learned Weighted Ensemble ('Learned W Ens') against each other and the baseline.

On **FiQA** (Table 2), all ensemble methods consistently outperformed the 'Selection' strategy. Compared to the 'Baseline', results were mixed: MiniLM ensembles underperformed, while weighted ensembles ('Fixed W', 'Learned W') generally improved performance for DistilBERT and DistilRoBERTa, surpassing the baseline. Simple averaging ('Avg Ens') was less effective than weighted methods.

On **Quora** (Table 3), weighted ensembles ('Fixed W', 'Learned W') showed strong and consistent performance gains over the 'Baseline' across all models. In contrast, 'Avg Ens' was unreliable, sometimes falling below the baseline. Weighted methods significantly outperformed 'Avg Ens', with 'Fixed W Ens' and 'Learned W Ens' achieving similar top-tier results.

Conclusion for RQ2: Ensembling strategy strongly impacts results. Weighted ensembles ('Fixed W', 'Learned W') consistently provide the best performance, significantly outperforming simple averaging ('Avg Ens') and often surpassing the monolithic 'Baseline', especially on the diverse Quora dataset. Simple averaging is less reliable. The strong, comparable performance of fixed and learned weights suggests the former offers a practical and effective approach.

4.3 Broader Implications

Our findings suggest that while monolithic models are strong baselines, performance gains may be achievable through more tailored approaches, though simple length-based specialization is insufficient. Specifically, ensembling multiple *lightweight* specialized models can outperform a single monolithic counterpart, offering a path to improved effectiveness potentially without drastically increasing model size. The success of simple fixed weighting highlights practical implementation benefits. However, the varying results between datasets underscore that the effectiveness of such specialization and ensembling strategies is likely context-dependent, influenced by factors like data scale and diversity.

4.4 Limitations and Future Work

This study's scope, constrained by resources, points to several avenues for future work. Our findings should be validated on **more diverse datasets and model architectures**, beyond the two datasets and three lightweight models tested here. The **specialization criterion (query length) was simplistic**; exploring richer features like query type, topic, or predicted difficulty could yield better results. Furthermore, the models' **limited input length restricts handling long documents**; integrating passage-based techniques (e.g., BERT-MaxP) is a necessary next step. We also acknowledge the **lack of empirical efficiency measurements**; future work should quantify the actual latency trade-offs. Finally, investigating **more sophisticated ensemble or learning-to-rank methods** beyond basic averaging and weighting could unlock further performance gains.

5 Conclusion

This work investigated the effectiveness of tailoring lightweight neural ranking models to query length. We found that dynamically selecting a single length-specialized model at inference time is unreliable and often degrades performance compared to a standard monolithic baseline. However, **ensembling these specialized models, particularly through simple fixed or learned weighting schemes, offers a promising approach to significantly boost retrieval effectiveness, often surpassing the monolithic baseline, especially on larger datasets.** The success of fixed weighting highlights a practical path forward. While query length alone may be too simple a criterion for specialization, the principle of combining signals from multiple "expert" lightweight models holds potential for building more effective and potentially resource-aware neural ranking systems. Future work exploring richer specialization criteria, advanced long-document handling, and diverse datasets will further illuminate the path beyond monolithic fine-tuning.

References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[2] Niranjan Balasubramanian and James Allan. 2010. Learning to select rankers. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Geneva, Switzerland) *(SIGIR '10)*. Association for Computing Machinery, New York, NY, USA, 855–856. doi:10.1145/1835449.1835650

[3] Claudio Carpineto and Giovanni Romano. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *ACM Comput. Surv.* 44, 1, Article 1 (Jan. 2012), 50 pages. doi:10.1145/2071389.2071390

[4] Gordon V. Cormack, Charles L.A. Clarke, and Stefan Buettcher. 2009. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and Development in Information Retrieval*. 758–759.

[5] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Ellen M. Voorhees, and Ian Soboroff. 2021. TREC Deep Learning Track: Reusable Test Collections in the Large Data Regime. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2369–2375. doi:10.1145/3404835.3463249

[6] Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. 2002. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Tampere, Finland) *(SIGIR '02)*. Association for Computing Machinery, New York, NY, USA, 299–306. doi:10.1145/564376.564429

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. doi:10.18653/v1/N19-1423

[8] Jiafeng Guo, Yinqiong Cai, Keping Bi, Yixing Fan, Wei Chen, Ruqing Zhang, and Xueqi Cheng. 2025. CAME: Competitively Learning a Mixture-of-Experts Model for First-stage Retrieval. *ACM Trans. Inf. Syst.* 43, 2, Article 35 (Jan. 2025), 25 pages. doi:10.1145/3678880

[9] Jiafeng Guo, Yinqiong Cai, Yixing Fan, Fei Sun, Ruqing Zhang, and Xueqi Cheng. 2022. Semantic Models for the First-Stage Retrieval: A Comprehensive Review. *ACM Trans. Inf. Syst.* 40, 4, Article 66 (March 2022), 42 pages. doi:10.1145/3486250

[10] Rolf Jagerman, Harrie Oosterhuis, Maarten de Rijke, et al. 2017. Query-Level Ranker Specialization.. In *LEARNER@ ICTIR*.

[11] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering.. In *EMNLP (1)*. 6769–6781.

[12] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 39–48. doi:10.1145/3397271.3401075

[13] Patrick Lewis, Barlas Oğuz, Wenhan Xiong, Fabio Petroni, Wen tau Yih, and Sebastian Riedel. 2021. Boosted Dense Retriever. arXiv:2112.07771 [cs.CL] https://arxiv.org/abs/2112.07771

[14] Bhaskar Mitra and Nick Craswell. 2018. An Introduction to Neural Information Retrieval. *Foundations and Trends® in Information Retrieval* 13, 1 (2018), 1–126. doi:10.1561/1500000061

[15] Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage Re-ranking with BERT. arXiv:1901.04085 [cs.IR] https://arxiv.org/abs/1901.04085

[16] Harshith Padigela, Hamed Zamani, and W. Bruce Croft. 2019. Investigating the Successes and Failures of BERT for Passage Re-Ranking. arXiv:1905.01758 [cs.IR] https://arxiv.org/abs/1905.01758

[17] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108 [cs.CL] https://arxiv.org/abs/1910.01108

[18] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models. arXiv:2104.08663 [cs.IR] https://arxiv.org/abs/2104.08663

[19] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. arXiv:2002.10957 [cs.CL] https://arxiv.org/abs/2002.10957

[20] Ji Xin, Rodrigo Nogueira, Yaoliang Yu, and Jimmy Lin. 2020. Early Exiting BERT for Efficient Document Ranking. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, Nafise Sadat Moosavi, Angela Fan, Vered Shwartz, Goran Glavaš, Shafiq Joty, Alex Wang, and Thomas Wolf (Eds.). Association for Computational Linguistics, Online, 83–88. doi:10.18653/v1/2020.sustainlp-1.11

A Appendix

Self-Assessment of Contributions

Serkan Akin: My primary contributions were in analysis and reporting. I performed the initial data analysis (BEIR datasets) and wrote the results analysis, interpreting performance metrics against the RQs. I also researched and authored the related work section. Additionally, I played a key role in shaping the final report's structure and narrative through writing and editing.

Elvira Antonogiannaki: I was mainly responsible for the writing and structuring of the final report, including drafting a substantial portion of the methodology section and reviewing the full manuscript for clarity and consistency. While I did not directly implement the codebase, I reviewed the implementation to ensure it aligned with the project's objectives and methodology, and ran preliminary experiments on a small corpus dataset (SciFact) to test components of the pipeline and support early-stage evaluation (the results are not included in the final paper). Additionally, I participated in all team meetings where key design and methodological decisions were made, and conducted background research — when necessary — to support these decisions.

Yiming Chen:I contributed to the early-stage exploration of the project's feasibility by finetuning Dense Passage Retrieval (DPR) models and evaluating their performance across different query lengths. These initial experiments played a key role in validating the feasibility of the project. In addition, I contributed to the writing of the report, I also supported the development of background content and helped refine the project's framing and motivation.

Yulin Chen: I was responsible for setting up and carrying out the experiments outlined in this paper. My notable contributions involved developing our data-processing pipeline, training the baseline and fine-tuned models, implementing ensembling strategies for evaluation, and incorporating statistical significance testing with the results we have gathered.

Link to Repository

The source code for this project can be found at our GitHub repository: https://github.com/yulinchen03/IR-rankingmodels.

Declaration of Generative AI Usage

During this project, the authors utilized generative AI tools to assist them with, brainstorming ideas, coding tasks and writing refinement. The authors directed and verified the core experimental work, analysis, and final manuscript, and they take full responsibility for the content.