Coarse-Tuning for Ad-hoc Document Retrieval Using Pre-trained Language Models

Atsushi Keyaki[†*], **Ribeka Keyaki**[‡]
[†]Dapartment of Social Data Science, Hitotsubashi University
2-1 Naka, Kunitachi, Tokyo, Japan
[‡]School of Computer Science, Tokyo University of Technology
1404-1 Katakuramachi, Hachioji City, Tokyo, Japan
*Corresponding author: a.keyaki@r.hit-u.ac.jp, keyakirbk@stf.teu.ac.jp

Abstract

Fine-tuning in information retrieval systems using pre-trained language models (PLM-based IR) requires learning query representations and query–document relations, in addition to downstream task-specific learning. This study introduces **coarse-tuning** as an intermediate learning stage that bridges pre-training and fine-tuning. By learning query representations and query–document relations in coarse-tuning, we aim to reduce the load of fine-tuning and improve the learning effect of downstream IR tasks. We propose **Query–Document Pair Prediction (QDPP)** for coarse-tuning, which predicts the appropriateness of query–document pairs. Evaluation experiments show that the proposed method significantly improves MRR and/or nDCG@5 in four ad-hoc document retrieval datasets. Furthermore, the results of the query prediction task suggested that coarse-tuning facilitated learning of query representation and query–document relations.

Keywords: pre-trained language model, neural information retrieval, coarse-tuning

1. Introduction

The advent of BERT (Devlin et al., 2019) has significantly improved the effectiveness of information retrieval systems using pre-trained language models (PLM-based IR) (Craswell et al., 2020a). However, simply applying fine-tuning on an IR dataset does not bring about a significant improvement (Lin et al., 2020), although fine-tuning is expected to achieve high effectiveness with additional minor training. High effectiveness requires costly fine-tuning, i.e., training with an IR-specific expensive network and/or training on a huge dataset such as MS MARCO (Bajaj et al., 2018). For example, CEDR (Robertson et al., 2019)'s fine-tuning involves training BERT for 100 epochs on a classification task, followed by training the proposed model for another 100 epochs. In PARADE (Li et al., 2020), fine-tuning involves training on the MSMARCO dataset, as well as another 36 epochs of training on the proposed model.

One possible reason for expensive fine-tuning is the difference in the nature of input data. The input for BERT's pre-training is natural language sentences. In contrast, the input for the IR task consists of pairs of two unbalanced data: `query`, which is a string of a few words, and `document`, which is dozens to hundreds of natural language sentences. Queries do not follow the grammar of natural language sentences (Barr et al., 2008) and thus conflict with models trained on natural language sentences[1]. It is highly possible that BERT's pre-training does not properly learn query representations that reflect the grammar of queries. Furthermore, the relation between a query and a document has a different nature from the relation between two sentences. That is, query words and their related words appear in candidate documents that match the query[2], while these words do not appear in irrelevant documents. Such query–document relations are also not learned in BERT's pre-training[3]. Therefore, fine-tuning in PLM-based IR requires learning query representations and query–document relations as well as downstream task-specific training for document ranking. In this way, the large gap in input data between pre-training and fine-tuning is the burden of fine-tuning, making fine-tuning more expensive.

We introduce an intermediate learning stage **coarse-tuning** that bridges pre-training for NLP tasks and fine-tuning for IR tasks. By learning query representations and query–document relations in coarse-tuning, fine-tuning focuses mainly on predicting the document's relevancy for the query, which helps to improve the effectiveness of fine-tuning at a low cost.

BERT's pre-training employs the Masked Language Model (MLM) for solving the cloze test and Next Sentence Prediction (NSP) for predicting sentence adjacency. We also use MLM to learn query

[1]The effectiveness is reduced when a POS tagger trained on natural language sentences is applied to queries (Barr et al., 2008; Ganchev et al., 2012; Keyaki and Miyazaki, 2017).

[2]Many studies that improve effectiveness by sampling words in documents to generate pseudo-queries (Ma et al., 2021a; Lee et al., 2019a; Chang et al., 2020).

[3]In fact, the result on Table 3 supported our claims.

LREC-COLING 2024, pages 3413–3421
20-25 May, 2024.

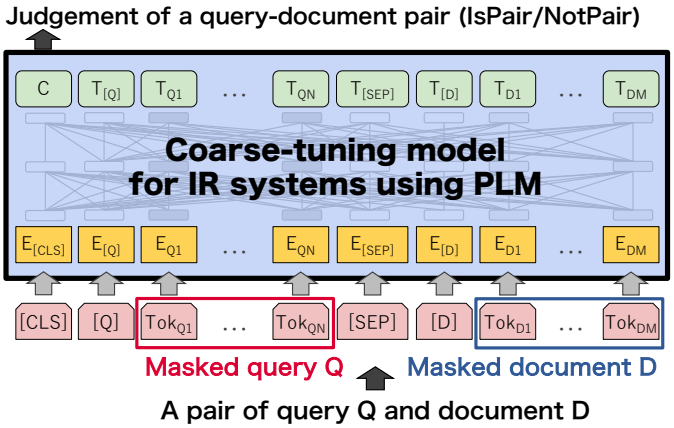

Figure 1: Architecture of the coarse-tuning model

representations. To learn query–document relations, we propose a **Query–Document Pair Prediction (QDPP)** that predicts the appropriateness of query–document pairs inspired by NSP.

Evaluation on four ad-hoc document retrieval datasets showed that applying coarse-tuning prior to fine-tuning statistically significantly improved MRR and/or nDCG@5. The results of query prediction suggested that the query representations and query–document relations were learned by coarse-tuning.

2. Proposed Method

Figure 1 shows an overview of the learning model for coarse-tuning. The architecture consists of multi-layer bidirectional Transformers, the same as BERT. In coarse-tuning, MLM and QDPP jointly learn query representations and query–document relations, respectively.

Model input: Since the learning model of coarse-tuning is specific to IR tasks, the input is a query–document pair. The query and document are tokenized and combined as a single sequence with special tokens (`[Q]`, `[D]`, `[CLS]`, `[SEP]`, and `[PAD]`). `[Q]` is inserted before the query tokens and represents the query. `[D]` is inserted before the document tokens and represents the document. The document tokens are truncated when exceeding the maximum token length.

Masked Language Model (MLM): We employ MLM to learn deep bidirectional query representations. A certain percentage of tokens are randomly masked (replaced with the `[MASK]` token). Learning is performed by predicting the original tokens of the masked tokens.

Query–Document Pair Prediction (QDPP): QDPP learns query–document relations through the task of predicting whether an input query–document pair is appropriate (`IsPair`) or inappropriate (`NotPair`).

Conditions of training data for coarse-tuning: The training data of coarse-tuning need to be (I) consisting of a large number of distinct queries and query–document pairs, and (II) preserving an appropriate relation between a query and a document. A straightforward resource for judging the appropriateness of query–document pairs is `qrels` (relevancy judgment) in an IR dataset. However, the amount of qrels is insufficient because typical qrels contain only dozens to hundreds of queries, with tens to hundreds of relevance judgments for each query. Thus, while qrels are a vital resource for learning how to rank documents in fine-tuning, they are not the optimal option to obtain a general representation of queries and query–document relations.

ORCAS (Open Resource for Click Analysis in Search) (Craswell et al., 2020a) is a click log dataset that satisfies both (I) and (II). ORCAS contains 19 million query–document pairs (10 million distinct queries, whose majority are keyword queries). Noise and inappropriate queries are removed from the vast amount of logs collected by Bing, and k-anonymization is applied to the remaining queries.

The query–document click relation is related to relevancy, and many studies achieved improvement by treating clicked documents as pseudo-relevant documents (Radlinski and Joachims, 2005; Huang et al., 2013; Wang et al., 2016). Therefore, we use query–document pairs in the clicked relation as appropriate query–document pairs.

During the generation of training instances, a certain percentage of query–document pairs from the set of appropriate query–document pairs (`IsPair`) are replaced with inappropriate documents (`NotPair`).

Learning procedure: Coarse-tuning focuses on learning query representations and query–document relations and does not aim at acquir-

ing general language representations. Therefore, we apply coarse-tuning to a pre-trained model where linguistic representations have already been learned. The procedure to learn the model for a downstream IR task is: (1) obtaining a pre-trained language model (PLM), (2) applying coarse-tuning (MLM and QDPP) using ORCAS, (3) fine-tuning on an IR dataset.

3. Experimental Evaluation

3.1. Experimental Setup

For evaluation on the downstream ad-hoc document retrieval datasets, the top 1,000 documents are retrieved using BM25 as the first stage retrieval, followed by re-ranking using the PLM-based IR methods. We compare the following six methods:

1. **`BM25`**
2. **`pre-trained`** Using a pre-trained model
3. **`coarse-tuned`** Applying coarse-tuning on a pre-trained model
4. **`fine-tuned (baseline)`** Applying fine-tuning on a pre-trained model
5. **`cont-pre+fine`** Applying continual pre-training with ORCAS documents on a pre-trained model followed by fine-tuning
6. **`coarse+fine (proposed)`** Applying coarse-tuning on a pre-trained model followed by fine-tuning

We employ a simple fine-tuning (Câmara and Hauff, 2020) to focus on verifying the effect of coarse-tuning. Specifically, fine-tuning is trained as a classification task that predicts the relevancy label of query–document pairs using the same model as coarse-tuning. The predicted probability of the relevant label is used as the document score.

In coarse-tuning, the additional dataset (ORCAS) is used for training. Even if the effectiveness of `coarse+fine (proposed)` improves, it's indistinguishable whether the improvement is from the coarse-tuning or simply increased training data. Therefore, we compare `coarse+fine (proposed)` to `cont-pre+fine`, where continual pre-training using ORCAS documents for 40 epochs is followed by fine-tuning. This comparison verifies the usefulness of coarse-tuning.

As for experimental environments, the pre-trained model is `prajjwal1/bert-tiny` (Bhargava, 2021) (L=2, H=128), the percentage of tokens to be masked in MLM was set to 0.15, the probability of generating a sample of `isPair` in QDPP was set to 0.5. We used PyTerrier (Macdonald and Tonellotto, 2020) and Hugging Face Transformers (Hugging Face, 2022) for development and evaluation. The maximum token length is 256. The batch sizes were set to 80 and 128 for coarse-tuning and fine-tuning, respectively. The optimizer used for learning was AdamW, which learning rate is 1e-3. The specs of the computer are CPU: AMD Ryzen 9 5900X 12-Core Processor, GPU: GeForce RTX 3060 VENTUS 2X 12G OC, memory: 128GB. Our proposed method took 20 hours for coarse-tuning (4 epochs) and 21 minutes for fine-tuning (3 epochs) on Robust04, which is within a practical cost range. However, the training time could be extended if larger models or more advanced fine-tuning methods are used.

3.2. Downstream Ad-hoc Document Retrieval Datasets

Robust04 (TREC, 2004) is often used in recent PLM-based IR evaluations because it consists of queries on which classical keyword match-based methods do not perform well. Robust04 contains 250 queries, 500,000 news articles, and 310,000 qrels. The number of qrels per query is larger than the standard IR datasets, indicating more dense judgments. The relevancy labels for qrels are `not relevant` (94.4%), `relevant` (5.3%), and `highly relevant` (0.3%). In the evaluation experiment, the minority label `highly relevant` was converted to `relevant`. We employ 4-fold cross-validation (CV) where 200 queries are used in training and 50 queries in evaluation.

We also evaluate the TREC Deep Learning (TREC-DL) Track datasets (Craswell et al., 2019, 2020b) (2-fold CV with 88 queries), GOV2 (TREC Terabyte Track) (TREC, 2004-2006) (3-foldCV with 150 queries), TREC-COVID (TREC, 2020) (5-fold CV with 50 queries) to verify the robustness of the proposed method. Robust04, GOV2, and TREC-COVID primarily consist of keyword queries, while more than two-thirds of the queries in TREC-DL are natural language queries. The language used in these datasets is English.

We report MRR, nDCG@5, 15, 30, and MAP as evaluation metrics. For verifying statistical significance, we used a paired two-sided t-test. Each sample is the effectiveness of each query; that is, the sample size is the number of queries. Note that we used the common coarse-tuned model for all experiments of the four datasets.

Preliminary experiments reported in Appendix A revealed that the optimal number of the ORCAS sampling rate is 8%, and epochs for coarse-tuning are 4, with fine-tuning epochs varying by dataset: 3 for Robust04[4], 1 for GOV2, 5 for TREC-COVID, and 4 for TREC-DL. These numbers are relatively small, and therefore the computational cost is not

[4]The first epoch was the most accurate when only fine-tuning was performed. This result suggests that coarse-tuning alleviates over-fitting. See Appendix A.3 for more details.

Table 1: **Evaluation on Robust04**: the symbols (*, †) indicates that there was a significant difference (p<0.01, p<0.05, respectively) compared to `fine-tuned (baseline)`.

	MRR	nDCG@5	nDCG@15	nDCG@30	MAP
`BM25`	0.541	0.371	0.356	0.332	0.214
`pre-trained`	0.201	0.100	0.109	0.119	0.125
`coarse-tuned`	0.227	0.115	0.124	0.136	0.143
`fine-tuned (baseline)`	0.548	0.377	0.359	0.361	0.279
`cont-pre+fine`	0.549	0.376	0.360	0.355	0.277
`coarse+fine (proposed)`	**0.597**†	**0.420***	**0.378**	**0.373**	**0.290**

Table 2: **Evaluation on GOV2, TREC-COVID, and TREC-DL**: the symbols (*, †, ‡) indicates that there was a significant difference (p<0.01, p<0.05, p<0.10) compared to `fine-tuned (baseline)`.

		MRR	nDCG@5	nDCG@15	nDCG@30	MAP
GOV2	`fine-tuned (baseline)`	0.514	0.327	0.327	0.308	**0.231**
	`coarse+fine (proposed)`	**0.568**†	**0.364**‡	**0.337**	**0.312**	0.230
TREC-COVID	`fine-tuned (baseline)`	0.873	0.821	0.791	0.789	0.739
	`coarse+fine (proposed)`	**0.940**†	**0.929***	**0.849**†	**0.821**†	**0.751**‡
TREC-DL	`fine-tuned (baseline)`	0.744	0.552	**0.521**	**0.517**	**0.463**
	`coarse+fine (proposed)`	**0.782**‡	**0.572**	0.508	0.514	0.452

high. The experiments in the next section report the effectiveness of the average score of five trials when optimal settings are used.

3.3. Evaluation of the Ad-hoc Document Retrieval Datasets

Table 1 shows the results on Robust04. The most effective method in all metrics was `coarse+fine (proposed)`. `coarse+fine (proposed)` improved 9% and 12% in MRR and nDCG@5, respectively, compared to `fine-tuned (baseline)`. The effectiveness of `cont-pre+fine` was roughly the same as `fine-tuned (baseline)`, suggesting that the improvement in `coarse+fine (proposed)` is not merely due to an increase in training data, but rather the effect of coarse-tuning. `pre-trained` and `coarse-tuned` that were not trained specifically for IR tasks performed poorly. These had even lower effectiveness than `BM25`, indicating the necessity of fine-tuning. In conclusion, fine-tuning improved the effectiveness with prior coarse-tuning.

As shown in Table 2, `coarse+fine (proposed)` outperformed `fine-tuned (baseline)` in MRR and nDCG@5 for GOV2, TREC-COVID, and TREC-DL. The effectiveness on TREC-DL did not improve as much as that on other datasets. In other words, `coarse+fine (proposed)` tended to show greater improvement with datasets consisting of keyword queries. Given that the majority of ORCAS queries are keywords, this suggests that having similar characteristics between coarse-tuning and fine-tuning data can enhance learning effectiveness.

All results with four datasets showed that `coarse+fine (proposed)` outperforms `fine-tuned (baseline)` with MRR and nDCG@5. It was confirmed that coarse-tuning improves the effectiveness of fine-tuning.

3.4. Evaluation of Query Representation and Query–Document Relations

To evaluate whether or not query representations and query–document relations were acquired by coarse-tuning, we conducted a task that predicts queries from given documents. Specifically, when creating a sequence from a document, we inserted `[MASK]` tokens at the position of the query tokens. Each model then predicts the `[MASK]` tokens for restoring the query. Since the average length of ORCAS queries is 3.27, the number of `[MASK]` tokens is set to 3 (Tok_{Q1}, Tok_{Q2}, Tok_{Q3}).

Table 3 is a list of top-5 predicted query tokens given the English Wikipedia article "Information retrieval"[5]. Both `pre-trained` and `fine-tuned (baseline)` predicted random tokens. Moreover, most of the first tokens Tok_{Q1} contain "##" despite the initial tokens. This suggests that the query representation and the query–document relation have been learned in neither BERT's pre-training nor its simple fine-tuning. In `coarse-tuned`, "how" and "what" appear in the first token Tok_{Q1} and "of" in the second token Tok_{Q2}, suggesting that some query representation has been learned. In addition, "data", "information", and "search" are words that appear in the input sequence, and "computer" and "citations" are words that appear in the article later than the input range. The result suggests that the query–document relation is also learned since the

[5] https://en.wikipedia.org/wiki/Information_retrieval

Table 3: Evaluation of query representation and query–document relation acquisition

	pre-trained			coarse-tuned			fine-tuned (baseline)			coarse+fine (proposed)		
	Tok_{Q1}	Tok_{Q2}	Tok_{Q3}	Tok_{Q1}	Tok_{Q2}	Tok_{Q3}	Tok_{Q1}	Tok_{Q2}	Tok_{Q3}	Tok_{Q1}	Tok_{Q2}	Tok_{Q3}
Top1	##lty	##lty	##lty	data	data	data	##ify	##ify	##ify	tv	tv	show
Top2	##nty	##tness	##tness	how	information	information	sure	##now	##now	show	show	tv
Top3	##gles	##rked	##rked	what	search	search	##now	sure	afford	oclc	chart	com
Top4	##tness	##gles	##rky	information	of	citations	guess	guess	guess	chart	oclc	chart
Top5	##rked	##nty	##lish	computer	.	wikipedia	##qui	##pass	sure	com	written	written

predicted query comprises words in the article and their related words.

On the other hand, the output for `coarse+fine (proposed)` is a set of tokens on a different topic from the input article, although the predicted tokens are consistent. A deeper analysis was conducted to interpret these results, as the result in Section 3.3 shows the effectiveness of `coarse+fine (proposed)`. We analyzed the terms that frequently appear in the "Information retrieval" article (such as "information", "search", "retrieval", "system", "query", etc., hereafter referred to as IR-related words) and the words in the training data for coarse-tuning and fine-tuning. As a result, we discovered that the words "tv", "show", "com", and "written", predicted in Table 3's `coarse+fine (proposed)`, co-occurred with IR-related words in the fine-tuning data (Robust04) with high probability. Moreover, some IR-related words ("retrieval" and "query") are less frequent in the training data, suggesting the possibility of overfitting (or a phenomenon close to catastrophic forgetting) due to limited contexts. Additionally, for the remaining predicted terms ("chart" and "oclc"), it was found that there is a relatively high probability of co-occurrence with some IR-related words in the coarse-tuning data (ORCAS queries) (such as "chart" with "rank" and "oclc" with "search"). Due to these factors, the search effectiveness and the results in Table 3 are not consistent. Therefore, using documents in a broader range of topics is beneficial when evaluating query-document relations.

Future work involves proposing a more effective coarse-tuning method and a fine-tuning method that does not lose query representations and query–document relations. Future work also includes observing behavior when larger BERT models and a more effective fine-tuning method are used.

4. Related Work

Although the mainstream research on PLM-based IR focuses on fine-tuning, there are also studies focusing on pre-training (Lee et al., 2019b; Ma et al., 2021a,b; Lee et al., 2019a; Chang et al., 2020; Ren et al., 2021; Wang et al., 2022; Ma et al., 2021c). Because Lee et al. (2019b); Ma et al. (2021a,b); Lee et al. (2019a); Chang et al. (2020) use pseudo-queries generated from documents for pre-training, these approaches differ from our study that learns query representations and query–document relations from real query–document pairs. PAIR (Ren et al., 2021) is designed specifically for passage retrieval and cannot be directly applied to the document retrieval we address in this study, making it different from our research. WebFormer (Wang et al., 2022) uses the structure of web documents for pre-training, which is different from our focus. Ma et al. (2021c) use pseudo-queries generated from the view that the anchor texts of hyperlinks and queries possess similar features. In contrast, we use real query-document pairs. One of the reasons we improved effectiveness with simple training is that using real queries allowed us to obtain higher-quality query representations. doc2query (Nogueira et al., 2019) and docTTTTTquery (Nogueira and Lin, 2019), which generate queries from documents, have in common with ours that they use real query–document pairs for training. However, since their purpose is vocabulary expansion, there is no interaction between queries and documents. These approaches are different from ours, where a query–document pair interacts with each other. ColBERT (Khattab and Zaharia, 2020), which delays query–document interactions to improve efficiency, is at the opposite end of the spectrum from our study.

Continual (further) pre-training (Sun et al., 2019; Gururangan et al., 2020; Zhu et al., 2021) using a corpus containing similar topics and a dataset of the downstream task is called domain-adaptive pre-training and task-adaptive pre-training, respectively. Ours differ from these approaches because we transform NLP tasks into IR tasks, which cannot be solved by simply training on similar datasets as we demonstrated in Section 3.3.

5. Conclusion

We proposed coarse-tuning for PLM-based IR for bridging pre-training and fine-tuning. Coarse-tuning consists of MLM for learning query representations and QDPP for learning query–document relations. We found coarse-tuning helps to improve the effectiveness of the downstream IR tasks. The acquisition of query representations and query–document relations was suggested in predicting queries from documents. These result suggests that coarse-tuning reduces the gap between pre-training and fine-tuning.

Ethical Considerations

- **Intended use** Our study aims to improve effectiveness with a small computational cost compared to existing methods. Therefore, our study benefits those who need a highly effective retrieval system with limited computing resources. No person or group is supposed to be harmed by our study.
- **Failure modes** The failure modes are that the operation of the search system stops, and the search system presents unnecessary results. As a result, system users suffer the inconvenience.
- **Biases** The most significant bias in search systems is position bias. Documents at the top of the search results are more exposed to users. It has nothing to do with failure modes.
- **Misuse potential** Not applicable.
- **Collecting data from users** Not applicable.
- **Potential harm to vulnerable populations** Not applicable.
- **Compute power** The computer we use does not have high specifications and has cheap learning costs (see Sec. 3.1).
- **The source code and the trained models** The source code is available at https://github.com/keyakkie/coarse-tuning.

Use of Existing Scientific Artifacts

- **ORCAS** ORCAS can be used for research use only and is under CC-BY 4.0. The language used is English.
- **Robust04** Robust04 is for research use only, which requires agreements to be filed with NIST. The language used is English.
- **GOV2 (TREC Terabyte Track dataset)** GOV2 (Web collection used TREC Terabyte Track) is for research use only, which requires agreements to be filed with the University of Glasgow. The language used is English.
- **TREC-COVID** TREC-COVID is for mining use only. The language used is English.
- **TREC Deep Learning (TREC-DL) Track datasets** TREC-DL Track datasets can be used for non-commercial research purposes only to promote advancement in the field of artificial intelligence and related areas and is under CC-BY 4.0. The language used is English.

Limitations

This study requires a pre-trained model and a large click log dataset. The behavior when using larger models or more sophisticated fine-tuning methods has yet to be examined.

Acknowledgments

This work was partially supported by the Japanese Society for the Promotion of Science Grant-in-Aid for Research Activity Start-up (#22K21303) and Grantin-Aid for Scientific Research (B) (#23H03686).

Bibliographical References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. arXiv:1611.09268.

Cory Barr, Rosie Jones, and Moira Regelson. 2008. The Linguistic Structure of English Web-Search Queries. In *Proc. of the EMNLP 2008*.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in NLI: Ways (not) to go beyond simple heuristics. arXiv:2110.01518.

Stefan Büttcher, Charles L. A. Clarke, and Ian Soborof. 2006. The TREC 2006 Terabyte Track. In *Proc. of the TREC 2006*.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pretraining Tasks for Embedding-based Large-scale Retrieval. In *Proc. of the ICLR 2020*.

Charles Clarke, Nick Craswell, and Ian Soborof. 2004. Overview of the TREC 2004 Terabyte Track. In *Proc. of the TREC 2004*.

Charles L. A. Clarke and Falk Scholer. 2005. The TREC 2005 Terabyte Track. In *Proc. of the TREC 2005*.

Nick Craswell, Emine Yilmaz Bhaskar Mitra, and Daniel Campos. 2021. Overview of the TREC 2020 deep learning track. arXiv:2102.07662.

Nick Craswell, Emine Yilmaz Bhaskar Mitra, Daniel Campos, and Ellen M. Voorhees. 2020a. Overview of the TREC 2019 deep learning track. arXiv:2003.07820.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020b. ORCAS: 18 Million Clicked Query-Document Pairs for Analyzing Search. arXiv:2006.05324.

Arthur Câmara and Claudia Hauff. 2020. Diagnosing BERT with Retrieval Heuristics. In *Proc. of the ECIR 2020*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of the NAACL 2019*.

Kuzman Ganchev, Keith Hall, Ryan McDonald, and Slav Petrov. 2012. Using Search-Logs to Improve Query Tagging. In *Proc. of the ACL 2012*.

Negin Ghasemi and Djoerd Hiemstra. 2021. BERT meets Cranfield: Uncovering the Properties of Full Ranking on Fully Labeled Data. In *Proc. of the ACL 2021*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *Proc. of the ACL 2020*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. In *Proc. of the CIKM 2013*.

Atsushi Keyaki and Jun Miyazaki. 2017. Part-of-speech Tagging for Web Search Queries Using a Large-scale Web Corpus. In *Proc. of the SAC 2017*.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proc. of the SIGIR 2020*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019a. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proc. of the ACL 2019*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019b. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proc. of the ACL 2019*.

Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2020. PARADE: Passage Representation Aggregation for Document Reranking. arXiv:2008.09093.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained Transformers for Text Ranking: BERT and Beyond. arXiv:2010.06467.

Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Xiang Ji, and Xueqi Cheng. 2021a. PROP: Pre-Training with Representative Words Prediction for Ad-Hoc Retrieval. In *Proc. of the WSDM 2021*.

Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Yingyan Li, and Xueqi Cheng. 2021b. B-PROP: Bootstrapped Pre-training with Representative Words Prediction for Ad-hoc Retrieval. In *Proc. of the SIGIR 2021*.

Zhengyi Ma, Zhicheng Dou, Wei Xu, Xinyu Zhang, Hao Jiang, Zhao Cao, and Ji-Rong Wen. 2021c. Pre-training for Ad-hoc Retrieval: Hyperlink is Also You Need. In *Proc. of the CIKM 2021*.

Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation in Information Retrieval using PyTerrier. In *Proc. of the ICTIR 2020*.

Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docTTTTTquery. https://cs.uwaterloo.ca/~jimmylin/publications/Nogueira_Lin_2019_docTTTTTquery-v2.pdf.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document Expansion by Query Prediction. arXiv:1904.08375.

Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. 2021. The Expando-Mono-Duo Design Pattern for Text Ranking with Pretrained Sequence-to-Sequence Models. arXiv:2101.05667.

Filip Radlinski and Thorsten Joachims. 2005. Query Chains: Learning to Rank from Implicit Feedback. In *Proc. of the KDD 2005*.

Navid Rekabsaz, Simone Kopeinik, and Markus Schedl. 2021. Societal Biases in Retrieved Contents: Measurement Framework and Adversarial Mitigation of BERT Rankers. In *Proc. of the SIGIR 2021*.

Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, and Ji-Rong Wen Haifeng Wang and. 2021. PAIR: Leveraging Passage-Centric Similarity Relation for Improving Dense Passage Retrieval. In *Proc. of the ACL 2021*.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 2019. CEDR: Contextualized Embeddings for Document Ranking. In *Proc. of the SIGIR 2019*.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification? arXiv:1905.05583.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. arxiv:1908.08962.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2020. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. *ACM SIGIR Forum*, 54:1–12.

Ellen M. Voorhees. 2004. Overview of the TREC 2004 Robust Retrieval Track. In *Proc. of the TREC 2004*.

Qifan Wang, Yi Fang, Anirudh Ravula, Fuli Feng, Xiaojun Quan, and Dongfang Liu. 2022. WebFormer: The Web-page Transformer for Structure Information Extraction. In *Proc. of the WWW 2022*.

Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Intent Based Relevance Estimation from Click Logs. In *Proc. of the SIGIR 2016*.

Qi Zhu, Yuxian Gu, Lingxiao Luo, Bing Li, Cheng Li, Wei Peng, Minlie Huang, and Xiaoyan Zhu. 2021. When does Further Pre-training MLM Help? An Empirical Study on Task-Oriented Dialog Pre-training. In *Proc. of the Second Workshop on Insights from Negative Results in NLP*.

Language Resource References

Prajjwal Bhargava. 2021. *prajjwal1/bert-tiny*. distributed via Hugging Face Hub. PID https://huggingface.co/prajjwal1/bert-tiny.

Craswell, Nick and Campos, Daniel and Mitra, Bhaskar and Yilmaz, Emine and Billerbeck, Bodo. 2020a. *ORCAS: Open Resource for Click Analysis in Search*. Microsoft Machine Reading Comprehension (MS MARCO) team hosted by Microsoft. PID https://microsoft.github.io/msmarco/ORCAS.html.

Nick Craswell and Bhaskar Mitra and Emine Yilmaz and Daniel Campos. 2019. *TREC 2019 Deep Learning Track: document ranking dataset*. Microsoft Machine Reading Comprehension (MS MARCO) team hosted by Microsoft. PID https://microsoft.github.io/msmarco/TREC-Deep-Learning-2019.

Nick Craswell and Bhaskar Mitra and Emine Yilmaz and Daniel Campos. 2020b. *TREC 2020 Deep Learning Track: document ranking dataset*. Microsoft Machine Reading Comprehension (MS MARCO) team hosted by Microsoft. PID https://microsoft.github.io/msmarco/TREC-Deep-Learning-2020.html.

Hugging Face. 2022. *Hugging Face Transformers Version 4.25.1*. Hugging Face. PID https://huggingface.co/docs/transformers/model_doc/bert.

Craig Macdonald and Nicola Tonellotto. 2020. *PyTerrier*. distributed via GitHub. PID https://github.com/terrier-org/pyterrier.

TREC. 2004. *TREC 2004 Robust Track dataset (Robust04)*. The National Institute of Standards and Technology (NIST). PID https://trec.nist.gov/data/t13_robust.html. The document set used in the track is the documents on TREC disks 4 and 5 and is distributed by NIST.

TREC. 2004-2006. *TREC Terabyte Track dataset (GOV2)*. The National Institute of Standards and Technology (NIST). PID https://trec.nist.gov/data/terabyte.html. The collection of Web data used in the track (GOV2) is distributed by the University of Glasgow on a hard disk drive.

TREC. 2020. *TREC-COVID Complete*. The National Institute of Standards and Technology (NIST). PID https://ir.nist.gov/trec-covid/data.html. The document set used in the track is COVID-19 Open Research Dataset (CORD-19) and is distributed on `https://allenai.org/data/cord-19`.

A. Preliminary Experiments

We evaluated the effectiveness of varying three parameters: the ORCAS sampling rate, the number of coarse-tuning epochs, and the number of fine-tuning epochs in a grid search on Robust04. The trend of the results was almost the same when each parameter was tuned independently and when the other parameters were varied. For this reason, we show the results using default values, 1, for parameters except for the tuning parameter, which means that each result differs from the values reported in Table 1.

A.1. ORCAS Sampling Rate

Because ORCAS contains a large number of query–document pairs, training was not complete when using the entire dataset. Therefore, we randomly sampled query–document pairs from the dataset from 1% to 10% in increments of 1%. As Table 4 indicates, effectiveness tended to improve with higher sampling rates in many settings.

Table 4: ORCAS sampling rate and its effectiveness

Sampling rate(%)	MRR	nDCG@5
1	0.507	0.357
2	0.487	0.339
3	0.526	0.337
4	0.523	0.352
5	0.500	0.364
6	0.499	0.359
7	0.546	0.357
8	**0.564**	**0.379**
9	0.524	0.375
10	0.537	0.364

Table 5: Number of epochs and effectiveness in coarse-tuning

# of epoch	MRR	nDCG@5
1	0.507	0.357
2	**0.523**	**0.365**
3	0.476	0.319
4	0.478	0.312
5	0.501	0.350

A.2. Number of Coarse-tuning Epochs

Up to 5 epochs were trained with coarse-tuning. In the single tuning, the second epoch showed the best effectiveness (see Table 5); however, in the majority of the settings, fourth epoch showed the best when the other parameters were varied.

A.3. Number of Fine-tuning Epochs

Up to 5 epochs were trained with fine-tuning. In the experiment that applied only fine-tuning, i.e., `fine-tuned (baseline)`, the first epoch achieved the highest effectiveness. Some prior studies of BERT-based rankers (Pradeep et al., 2021; Ghasemi and Hiemstra, 2021; Rekabsaz et al., 2021) have also reported saturating in a few epochs, which is relatively smaller than other NLP tasks. This shows that fine-tuning in an IR task is prone to over-fitting. It suggests that there is a large gap between pre-training and fine-tuning, namely, the difference in the nature of input data that makes it difficult to learn IR task-specific representation throughout epochs. In contrast, when combined with coarse-tuning, the third epoch in fine-tuning showed the highest effectiveness in most settings. This result suggests that coarse-tuning reduces the gap between pre-training and fine-tuning and supports learning in fine-tuning, thus alleviating over-fitting.

Table 6: Number of epochs and effectiveness in fine-tuning

# of epoch	MRR	nDCG@5
1	**0.507**	**0.357**
2	0.490	0.338
3	0.488	0.338
4	0.467	0.324
5	0.477	0.327