Incorporating Relevance Feedback for Information-Seeking Retrieval using Few-Shot Document Re-Ranking

Tim Baumgärtner,[1] **Leonardo F. R. Ribeiro,**[1*] **Nils Reimers,**[2] **Iryna Gurevych**[1]
[1]Ubiquitous Knowledge Processing Lab (UKP Lab),
Department of Computer Science and Hessian Center for AI (hessian.AI),
Technical University of Darmstadt
[2]cohere.ai
www.ukp.tu-darmstadt.de

Abstract

Pairing a lexical retriever with a neural re-ranking model has set state-of-the-art performance on large-scale information retrieval datasets. This pipeline covers scenarios like question answering or navigational queries, however, for information-seeking scenarios, users often provide information on whether a document is relevant to their query in form of clicks or explicit feedback. Therefore, in this work, we explore how relevance feedback can be directly integrated into neural re-ranking models by adopting few-shot and parameter-efficient learning techniques. Specifically, we introduce a kNN approach that re-ranks documents based on their similarity with the query and the documents the user considers relevant. Further, we explore Cross-Encoder models that we pre-train using meta-learning and subsequently fine-tune for each query, training only on the feedback documents. To evaluate our different integration strategies, we transform four existing information retrieval datasets into the relevance feedback scenario. Extensive experiments demonstrate that integrating relevance feedback directly in neural re-ranking models improves their performance, and fusing lexical ranking with our best performing neural re-ranker outperforms all other methods by 5.2% nDCG@20.[1]

1 Introduction

User queries can be categorized as navigational (retrieving a specific document), transactional (retrieving a website to perform a particular action) or informational (Broder, 2002). For information-seeking queries, users might want to learn about a new topic or might be unfamiliar with the search domain. Therefore they potentially do not use common keywords of the domain which decreases performance (Furnas et al., 1987). Furthermore, they might want to find complementary information from diverse sources or consider different aspects of a topic (Clarke et al., 2008). Lastly, information-seeking queries can also be used to keep up with the latest developments on a topic.

Concretely, these queries are encountered during scientific literature review (Voorhees et al., 2021; Dasigi et al., 2021), when looking for news and background information (Soboroff et al., 2018), during argument retrieval, (Bondarenko et al., 2021) or in the legal context for case law retrieval (Locke and Zuccon, 2018).

Formulating effective queries to satisfy the complex information need in these scenarios is difficult. On the contrary, a user can easily judge whether a document is relevant to their query. Therefore, information obtained from the user when interacting with the search results, known as *relevance feedback*, can be used in the search. This can be obtained implicitly from click logs (Joachims, 2002) or explicitly by asking users whether a document is relevant (Rocchio, 1971). We focus on explicit feedback because it is clean compared to implicit feedback and existing information retrieval datasets can be transformed into this scenario. In both settings, the amount of feedback is limited, since users will provide feedback only on a few documents.

Incorporating relevance feedback in information retrieval (IR) systems is well-established for lexical retrieval (Rocchio, 1971; Lavrenko and Croft, 2001; Zhai and Lafferty, 2001). These systems incorporate the feedback by expanding the query with terms extracted from relevance feedback documents. While these approaches can alleviate the lexical gap, they inherently struggle with semantics because they represent text as a bag of words. Additionally, lexical query expansion methods have the disadvantage that their latency increases with the number of query terms (Wu and Fang, 2013).

To mitigate these issues, neural retrieval and re-ranking methods have been proposed and recently

[*]Now affiliated with Amazon Alexa AI.
[1]The code is available at https://github.com/UKPLab/incorporating-relevance

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 8988–9005
December 7-11, 2022

outperformed lexical retrieval (Gillick et al., 2019; Nogueira and Cho, 2019; Karpukhin et al., 2020; Khattab and Zaharia, 2020). State-of-the-art retrieval results are obtained in a two-stage setup: First, an efficient and recall-optimized retrieval method (e.g. dense or lexical retrieval) retrieves an initial set of documents. Subsequently, a neural re-ranker optimizes the rank of the documents. However, there exists no neural re-ranking model that directly incorporates relevance feedback.

To this end, we explore how relevance feedback can directly be integrated into neural re-ranking models. This is difficult because state-of-the-art models have millions of parameters and require a large amount of training data, while only a limited amount of relevance feedback per query is available. We make use of recent advances in parameter-efficient fine-tuning (Houlsby et al., 2019; Ben Zaken et al., 2022) and few-shot learning (Snell et al., 2017; Finn et al., 2017) to address the challenges of model re-usability and learning from limited data. Concretely, we present a kNN approach that re-ranks documents based on their similarity to the feedback documents. We further propose to fine-tune a re-ranking model from only the relevance feedback for each query. We explore the effectiveness of our approach with a varying number of feedback documents and evaluate its computational efficiency. To evaluate our models, we transform four existing IR datasets into the re-ranking with relevance feedback setup. Our final model combines the strengths of lexical and neural re-ranking using reciprocal rank fusion (Cormack et al., 2009).

In summary, our contributions are as follows:

- We propose a few-shot learning task for information retrieval, specifically adopting the two-stage retrieve and re-rank settings to incorporate relevance feedback, both in the retrieval as well as in the re-ranking.
- We outline retrieval scenarios for the task and how to transform existing IR datasets into the few-shot retrieve and re-rank setup.
- We present novel re-ranking methods that directly incorporate relevance feedback leveraging few-shot learning and parameter-efficient techniques. We evaluate their efficiency and demonstrate their effectiveness through extensive experiments and across different datasets.

2 Related Work

2.1 Information Retrieval Approaches

Traditionally, lexical approaches have been used for IR, such as TF-IDF and BM25 (Robertson and Zaragoza, 2009). However, these systems cannot model lexical-semantic relations between query and document (the document and query are treated as bag of words) and suffer from the lexical gap (Berger et al., 2000), e.g., when synonyms are used.

Recently, dense retrieval methods have shown promising results, outperforming lexical approaches (Gillick et al., 2019; Karpukhin et al., 2020; Khattab and Zaharia, 2020). Contrary to lexical systems, they can discover semantic matches between a query and a document, thereby overcoming the lexical gap. Dense retrieval methods learn query and document representations in a shared, high-dimensional space. This is enabled by large-scale pre-training (Devlin et al., 2019) and training on IR datasets of considerable size (Nguyen et al., 2016; Kwiatkowski et al., 2019). After training, the model computes a document index holding a representation for each document in the corpus. At inference, a query representation is compared to each document vector using maximum inner product search (Johnson et al., 2021).

However, applying dense retrieval to our setup is not practical. We aim to fine-tune the model for every query, therefore, the precomputed document index would become out of sync with the model and might not yield optimal results (Guu et al., 2020). Since the document index is very large, re-encoding it would create an unreasonable computational overhead. Thus, we do not experiment with dense retrieval models that rely on a precomputed document index.

Similar to dense retrieval, neural re-ranking models have profited from pre-training and training on large datasets. The predominant approach is to use a Cross-Encoder (CE) model that takes both query and document as input to directly compute a relevance score. Contrary to dense retrieval models, this enables direct query-document interactions. Since this approach does not allow to pre-compute representations and is compute-intensive, it is generally paired with a more efficient first-stage retrieval method (dense or lexical) and subsequently applied to the top retrieved documents. Particularly combined with lexical retrieval methods, neural re-ranking yields state-of-the-art performance (Thakur et al., 2021).

Dataset	Domain	Docs	Doc. Length	Queries	Q. Length	Judgments
Robust04 (Voorhees, 2004)	News	528k	476.40	148	16.76	1287.14 (±501)
TREC-Covid (Voorhees et al., 2021)	Biomedical	191k	158.87	50	10.96	1370.36 (±323)
TREC-News (Soboroff et al., 2018)	News	595k	686.65	34	12.03	258.85 (±82)
Webis-Touché (Bondarenko et al., 2021)	Debates	383k	289.34	49	6.67	49.76 (±7)

Table 1: Datasets used for the few-shot re-ranking task. Length: average number of words. Judgments: average number (and standard deviation) of relevant and non-relevant judged documents per query. The datasets have been filtered to only include queries with a minimum number of relevant and non-relevant documents.

2.2 Relevance Feedback

Relevance feedback has mostly been integrated into IR systems by modifying the query using the feedback documents and subsequently performing a second round of retrieval. Rocchio (1971) propose a linear combination of the vectors of the query, the relevant and non-relevant feedback documents to obtain a new query vector, which is more similar to the relevant documents. Another approach is to use language models of the query and documents to obtain new terms (Lavrenko and Croft, 2001; Zhai and Lafferty, 2001). Recently, Naseri et al. (2021) use the similarity between contextualized query and document word embeddings to extract terms for query expansion. Similarly, Zheng et al. (2020) use BERT to obtain document chunks for expansion and subsequently compute the relevancy by summing over chunk-document relevance. While these works leverage advances in pre-trained language modeling for selecting query terms, they eventually rely only on lexical retrieval, potentially missing semantic matches in the document collection. While we also use lexical retrieval with query expansion for the second stage, we additionally update a re-ranking model based on the relevance feedback and employ it on the second stage retrieval results.

Other works directly incorporate relevance feedback into neural retrieval. Ai et al. (2018) train a model that sequentially encodes the top document representations from the first stage retrieval. The documents are subsequently re-ranked using an attention mechanism between the final and intermediate representations of the model. Yu et al. (2021) further fine-tune the query encoder of a dense retrieval model to additionally take the top documents from a first retrieval stage as input. While these works directly incorporate first-stage retrieval documents into their model, they require large annotated datasets to train their models. Furthermore, adding the feedback documents to the input is sub-optimal due to large memory requirements of transformer models with growing input size. Our approach overcomes this by using the relevance feedback to update the model parameters instead of providing it as input.

Most similar to our work, Lin (2019) propose to learn a re-ranker using machine learning classifiers (logistic regression and support vector machines) based on lexical features from the top and bottom retrieved documents. They show that this simple approach improves over query expansion and neural approaches like NPRF (Li et al., 2018). In contrast to our work, they use pseudo-relevance feedback and simple classification approaches as a re-ranking model. Moreover, we use explicit relevance feedback since the automatic selection of non-relevant documents is challenging. Depending on the query and document collection the number of relevant documents varies significantly. For one query there might only be few relevant documents in which case irrelevant documents could be selected from higher ranks. Another query might return a large set of relevant documents in which case pseudo-irrelevant documents would actually need to be selected from lower ranks. User feedback on the other hand does not have this disadvantage. However, since users will only give feedback on limited documents, the models used by Lin (2019) cannot be trained from explicit feedback. Therefore, we opt for few-shot learning combined with pre-trained re-ranking models. Furthermore, the user-selected documents also provide a form of interpretability to the re-ranking model.

In summary, using pseudo and explicit relevance feedback in lexical models via query expansion has shown to improve retrieval performance. Furthermore, neural retrieval and re-ranking models have shown promising results, outperforming lexical methods. While there exists related work that combines neural models with query expansion, they are applied to pseudo-relevance feedback and use state-of-the-art models only for determining query expansion terms. Other methods are limited in the amount of feedback and require large training

datasets for fine-tuning. In this work, we leverage few-shot learning techniques to directly update a re-ranking model based on explicit feedback.

3 Datasets

Large-scale IR datasets mostly target use cases where a user has a less complex information need, e.g., looking for a factoid answer (Nguyen et al., 2016; Kwiatkowski et al., 2019; Zhang et al., 2021). These are usually sparsely annotated, i.e. there is only a single (or few) judged relevant documents per query. However, for our information-seeking use case, we are interested in queries where many relevant documents exist. Therefore, we select datasets where a large set of relevant documents per query are judged. Further, the datasets should target suitable use cases containing queries that have a diverse set of relevant documents. For example, the query "What is the origin of Covid-19" from TREC-Covid, has relevant documents about the geographical location of the first cases, the genetic origins of the virus, and animals that likely have transmitted the disease to humans.

Specifically, we consider Robust04 (Voorhees, 2004), TREC-Covid (Voorhees et al., 2021), TREC-News (Soboroff et al., 2018), and Webis-Touché (Bondarenko et al., 2021). An overview of all datasets with their statistics is provided in Table 1.[2] We transform these datasets into the few-shot re-ranking setup by including only queries with at least 32 judged relevant and non-relevant documents in the BM25 top 1000 results with the query. Any queries with fewer judged documents are discarded because they provide little evaluation power, because we remove the feedback documents from the evaluation. Moreover, this filter ensures that enough judged documents are present for a robust evaluation.

For our experiments, we create training, validation and test splits in a 3:1:1 ratio, by randomly assigning each query to one set. We further conduct three random shuffles over the assignment of a query into the training, validation, and test set. We report the averaged results over the shuffles.

4 Task Setup

To incorporate relevance feedback in any retrieval process, a multi-stage approach is required. We propose a multi-phase task setup which is visualized in Figure 1. In **Phase 1**, the relevance feedback

[2]More details on the datasets in Appendix A.

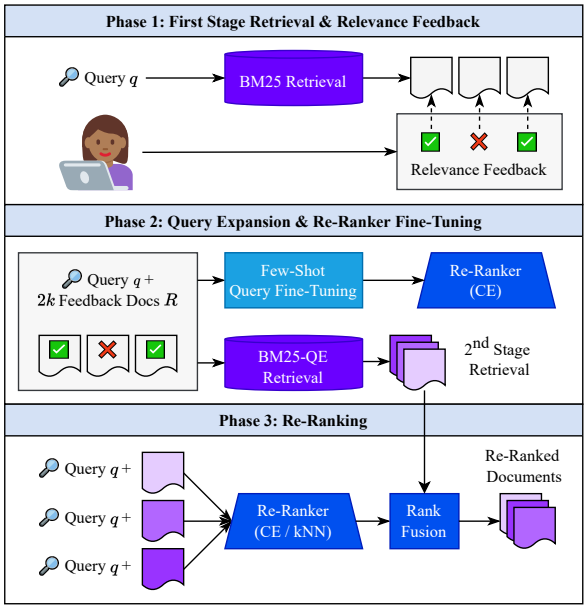

Figure 1: The three phases of our proposed few-shot retrieve and re-rank setup. **Phase 1**: Documents are retrieved using the query q, and relevance feedback is obtained from a user. **Phase 2**: The query q and feedback documents R are used for query expansion and the second round of retrieval. Further, a re-ranking model is fine-tuned using the user-selected feedback documents. **Phase 3**: The documents are re-ranked using the fine-tuned re-ranker, obtaining the final document ordering. To improve performance, the ranking from the re-ranker and the second phase are be fused.

is collected from the user after a first retrieval. The selected documents refine the information need and provide additional insight into what is relevant to the query. In **Phase 2**, the feedback is processed and a second retrieval is conducted while the re-ranking model is trained on the selected feedback documents. This phase returns documents that are more relevant to the user's information need. Ultimately, in **Phase 3**, the documents obtained previously are re-ranked based on the tailored re-ranker.

Specifically, in **Phase 1**, an initial retrieval is conducted with the query q against the document collection. For lexical retrieval, we use BM25 as it is robust in a zero-shot setting on a diverse set of domains (Thakur et al., 2021). Next, we select the top $k \in \{2, 4, 8\}$ relevant and non-relevant documents from the first-stage retrieval according to the judgments in the dataset, i.e., there are $2k$ documents selected per query.

We refer to these documents as *feedback documents* R. By selecting the top judged and retrieved documents, this process simulates a user provid-

ing relevance feedback.[3] For our evaluation, we remove the feedback documents from the relevance judgments (i.e. we use the *residual collection*, (Salton and Buckley, 1990) in order to evaluate the ability of the models to rank non-selected documents higher.

In **Phase 2**, we use the $2k$ feedback documents for query expansion and a second retrieval step. We extract e terms per relevant feedback document and append them to the query resulting in the expanded query for second-stage retrieval.[4] Furthermore, the feedback documents are used to fine-tune a re-ranking model. Starting from a common base model, a new model is fine-tuned for every query. To exploit the small number of feedback documents most effectively, we employ few-shot learning when fine-tuning the re-ranker.

Finally, in **Phase 3**, the documents are scored using the query-specific re-ranker from the second phase. Additionally, the ranking from BM25-QE can be incorporated in the final document ranking. We experiment with different models and settings, details are described in §5.

The re-ranking could also be performed on the documents from the first-stage retrieval. However, since the feedback documents are available and query expansion generally improves recall (which is important for the re-ranking performance), we chose to not experiment with re-ranking the first-stage retrieval documents. This also improves the evaluation, because all models re-rank the same set of documents.

Evaluation Metrics. To measure the ranking performance we use nDCG@20 (Järvelin and Kekäläinen, 2000) implemented by PYTREC_EVAL (Gysel and de Rijke, 2018). This metric considers graded relevance labels. We chose the cut-off at 20 to take the large number of relevant documents per query into account. Beyond ranking performance, we also focus on retrieval/re-ranking latency and parameter efficiency. The response time of IR systems is generally crucial for user satisfaction (Schurman and Brutlag, 2009). Therefore, we evaluate the time for retrieval, query expansion, fine-tuning, and re-ranking. Since we fine-tune a model per query, the memory footprint of the model should be small. This allows keeping many models in memory at the same time or quickly reloading a model whenever a user revisits a query.

[3]We also experimented with selecting judged documents randomly but preliminary experiments showed that this generally leads to worse performance.

[4]We also experimented with negatively weighing terms from non-relevant documents. However, we find that this generally hurts performance.

5 Methods

5.1 BM25 Query Expansion

For the second-stage retrieval, we expand the original query q with terms e obtained from the relevant feedback documents. We experiment with a varying number of expansion terms $e \in \{4, 8, 16, 32, 64\}$ and also use all terms in the document for expansion which we refer to as *all*. We do so by using Elasticsearch's MoreLikeThis feature,[5] which extracts terms according to their TF-IDF score. For retrieval, the query and the extracted terms are combined, and the documents are scored according to BM25. This setup follows the query expansion technique described in Rocchio (1971). The ranking produced by BM25 query expansion (BM25-QE) serves as the lexical baseline in our experiments.

5.2 Re-ranker

In this section, we detail the different approaches employed for document re-ranking: kNN, Cross-Encoder, and Rank Fusion.

5.2.1 kNN

The kNN approach is based on a dense retrieval model that computes a high-dimensional, semantic text representation. Specifically, we use the transformer-based MiniLM (Wang et al., 2020b) model that was fine-tuned on a diverse set of training datasets.[6]

We use the 6-Layer model since its counterpart with 12 layers only provides marginally better performance albeit requiring twice the compute.

To obtain a document score s_i, we compute the similarity between the query q and the document d_i and add the sum of similarities between the relevant feedback documents $d_j \in R^+$ and d_i. We use cosine-similarity as similarity function f. This is expressed in Equation 1.

$$s_i = f(d_i, q) + \sum_{d_j \in R^+} f(d_i, d_j) \qquad (1)$$

The kNN setup resembles Prototypical Networks (Snell et al., 2017), however, instead of having a

[5]Elasticsearch: MoreLikeThis

[6]https://discuss.huggingface.co/t/train-the-best-sentence-embedding-model-ever-with-1b-training-pairs/7354

single, averaged point in vector space representing a class, we have $k+1$ points (all relevant feedback documents and the query). In this setting, the model weights are not updated, instead, we use the document and query encodings for finding similar documents.

5.2.2 Cross-Encoder (CE)

For re-ranking with a Cross-Encoder, we employ the 6-Layer MiniLM model fine-tuned on MS MARCO (Hofstätter et al., 2020). We experiment with zero-shot, query fine-tuning, and meta-learning approaches.

Zero-Shot. As a baseline, we do not perform any fine-tuning and re-rank the documents with the pre-trained model. Zero-shot only refers to not fine-tuning the re-ranking model, however, we still re-rank the documents obtained with query expansion for comparability reasons.

Query Fine-Tuning. We update the re-ranker using few-shot supervised learning with the $2k$ feedback documents. We optimize the Binary Cross-Entropy and use the validation set to determine the learning rate and the number of training steps that perform best on average according to the nDCG@20 score. We refer to this as *CE Query-FT*.

Meta-Learning. In order to optimize the model for quick adaption to new queries, we also explore using model-agnostic meta-learning (MAML) (Finn et al., 2017). Meta-learning is generally defined over a set of tasks (as opposed to a set of training samples). Therefore, we treat each query with the respective feedback documents as its own task. This is reasonable since we model the relevance of a document in the context of the query. The training process consists of two stages: (1) First, the model g is optimized on the training dataset. Each batch consists of two tasks T_1 and T_2, each comprising a query and the respective $2k$ feedback documents. The model parameters θ are updated using T_1 optimizing the Binary Cross-Entropy on the feedback documents with learning rate α. We obtain new parameters θ' from this step. We show this formally in Equation 2 for a single step.

$$\theta' = \theta - \alpha \nabla_\theta \mathcal{L}(g_\theta; T_1) \tag{2}$$

Subsequently, the new parameters are evaluated on their ability to adapt to the second task T_2 by computing the loss of the predictions made by the model $g_{\theta'}$. By backpropagating through this entire process (i.e. computing the gradients w.r.t. to θ), the original parameters of the model are optimized:

$$\theta'' = \theta - \alpha \nabla_\theta \mathcal{L}(g_{\theta'}; T_2) \tag{3}$$

Intuitively, the loss in Equation 3 will be low, if the parameters θ' can quickly adapt to T_2. We refer the reader to Finn et al. (2017) for a more detailed overview of the training process using MAML. In our training process, we only use a single task, i.e. one query with its respective feedback documents, per step due to the limited amount of training data. We find the hyperparameters according to the zero-shot performance on the validation dataset. (2) Once the MAML training concludes, the model is updated per query as detailed in the Query Fine-Tuning paragraph. We call this method *CE MAML + Query FT*.

Parameter Efficiency. For all Cross-Encoder methods we only update the bias layers as proposed by Ben Zaken et al. (2022). This keeps the number of tunable parameters and the memory footprint of the models very small. Using this method only 0.11% of the parameters are updated. Compared to adapters (Houlsby et al., 2019), tuning the biases is advantageous because the parameters are already tuned and not randomly initialized.

5.2.3 Rank Fusion

We also investigate merging the rankings produced by BM25-QE and the neural re-ranking model using Reciprocal Rank Fusion (RRF) (Cormack et al., 2009). The final ranking is computed according to Equation 4, where s_i is the fused score of document d_i, h is the ranking function returning the rank of a document and c is a constant decreasing the impact of the top-scored documents.[7]

$$s_i = \sum_{h \in H} \frac{1}{c + h(d_i)} \tag{4}$$

This approach has the advantage of being agnostic to the relevance scores assigned to the documents by the models because it only uses their rankings. Using the relevance scores directly is problematic when the scores of the models are in different ranges.[8] Intuitively, RRF can leverage diverse rankings to improve the result. Furthermore,

[7] We leave the constant at the default value of 60.

[8] E.g., BM25 can produce large scores per document as it is a sum of scores, while binary classification models like Cross-Encoder models produce scores between 0-1.

e	$k=2$	$k=4$	$k=8$	Avg.
4	0.6187	0.6300	0.6566	0.6351
8	**0.6280**	**0.6414**	0.6721	**0.6472**
16	0.6195	0.6400	**0.6736**	0.6444
32	0.6039	0.6209	0.6477	0.6242
64	0.5597	0.5729	0.5843	0.5723
all	0.5723	0.5771	0.5828	0.5774

Table 2: Recall@1000 results on the test set with varying number of expansion terms e from each relevant document. Results are averaged over the shuffles.

it will rank documents higher that are strongly preferred by one ranking model than documents that are weakly preferred by multiple models.[9]

6 Results

6.1 2nd Stage Retrieval: Query Expansion

We report recall@1000 results of the second stage retrieval with varying number of expansion terms e in Table 2. Note that by increasing e the performance increases, reaching a maximum at $e = 8$. However, when further increasing e, the recall drops. We observe qualitatively that extracting more terms per document also includes more non-specific terms or even stop words which hurt performance. Based on the recall@1000 performance on the validation set (see Appendix I) we use the documents obtained by extracting $e = 16$ terms for the final re-ranking step. In this work, we do not focus on the first stage retrieval. For completeness, we report the results in Appendix B.

6.2 Re-Ranking Performance

We report the nDCG@20 ranking performance in Table 3 and additional zero-shot baselines in Appendix C. We first note that increasing the amount of relevance feedback k generally improves performance. Furthermore, we observe that BM25-QE already performs well. Neither the kNN approach, nor the Cross-Encoder zero-shot and Query FT, nor the wide variety of zero-shot models are able to outperform BM25-QE, except on TREC-Covid. We note a superior performance on Webis-Touché, although this task is the most challenging for neural models in our test suite. This agrees with related work that indicates that BM25 beats all other methods on this task (Thakur et al., 2021). When looking at the CE experiments, we observe incremental

[9]For example, if $h_0(d_0) = 5$, $h_1(d_0) = 15$, $h_0(d_1) = 10$ and $h_1(d_1) = 10$ then $s_0 > s_1$.

	Robust	Covid	News	Touché	Avg.
BM25-QE					
$k=2$	0.4480	0.5632	0.3846	0.2602	0.4140
$k=4$	0.4843	0.6079	0.3877	0.2558	0.4339
$k=8$	0.5568	0.6606	0.4049	**0.2982**	0.4801
Avg.	0.4964	0.6106	0.3924	0.2714	0.4427
kNN					
$k=2$	0.4259	0.6736	0.3492	0.1646	0.4033
$k=4$	0.4342	0.6789	0.3539	0.1697	0.4092
$k=8$	0.4698	0.7069	0.3925	0.1904	0.4399
Avg.	0.4433	0.6865	0.3652	0.1749	0.4175
CE Zero-Shot					
$k=2$	0.3937	0.6917	0.2955	0.1731	0.3885
$k=4$	0.4185	0.7018	0.3189	0.1767	0.4040
$k=8$	0.4335	0.7150	0.3285	0.1799	0.4142
Avg.	0.4152	0.7028	0.3143	0.1766	0.4022
CE Query-FT					
$k=2$	0.4375	0.6833	0.2942	0.1887	0.4009
$k=4$	0.4786	0.7182	0.3463	0.2080	0.4378
$k=8$	0.5376	**0.7677**	0.3645	0.1975	0.4668
Avg.	0.4846	0.7231	0.3350	0.1981	0.4352
CE MAML + Query FT					
$k=2$	0.4529	0.7129	0.2526	0.2212	0.4099
$k=4$	0.5079	**0.7498**	0.3358	0.2292	0.4557
$k=8$	0.5572	0.7449	0.3557	0.2201	0.4695
Avg.	0.5060	0.7359	0.3147	0.2235	0.4450
Rank Fusion: kNN & BM25-QE					
$k=2$	0.4635	0.6903	0.3783	0.2263	0.4396
$k=4$	0.5020	0.6858	**0.4228**	0.2438	0.4636
$k=8$	0.5574	0.7470	**0.4359**	0.2744	0.5037
Avg.	0.5076	0.7077	**0.4123**	0.2482	0.4689
Rank Fusion: CE MAML + Query FT & BM25-QE					
$k=2$	**0.5164**	**0.7269**	**0.3934**	**0.2670**	**0.4759**
$k=4$	**0.5576**	0.7449	0.4084	**0.2701**	**0.4953**
$k=8$	**0.6380**	0.7489	0.4148	0.2809	**0.5207**
Avg.	**0.5707**	**0.7402**	0.4055	**0.2727**	**0.4973**

Table 3: nDCG@20 test set results averaged over three seeds with a varying number of feedback documents (k). In bold, the best performing model, the runner-up is underlined.

performance increases when the relevance feedback is integrated. CE zero-shot is outperformed by query fine-tuning, which is subsequently outperformed when MAML training is added. This shows that our proposed direct integration of relevance feedback in the model is effective and that the parameters obtained by MAML training are better able to adapt to new queries given the relevance feedback. This method also slightly outperforms BM25-QE.

Finally, combining the rankings of the lexical retrieval and neural re-ranker is particularly effective. While different methods excel at each dataset (e.g. BM25 on Webis-Touché or neural models on

TREC-Covid), the rank fusion is able to mitigate the weaknesses of one model successfully. Moreover, combining two rankings often outperforms the single ranking, showing that query expansion and neural re-ranking are highly complementary.[10] We analyze the intersection of the top documents between BM25-QE and the two re-rankers. We find that in more than 50% of the queries in the test set, BM25-QE and the re-ranking model only agree on 5 or fewer documents in the top 20.[11]

6.3 Re-Ranking Ablations

To gain further insights into where our performance improvements are coming from, we conduct a series of ablation studies, reported in Table 4.

First, we ablate the influence of query expansion and the feedback documents on lexical retrieval. We retrieve only using the query and remove the feedback documents from the retrieval and evaluation, i.e. we use the residual collection, even though the feedback documents are not used. From the first section of Table 4 we can observe a large performance drop. This shows that BM25-QE is successfully able to exploit the feedback documents and retrieve more relevant documents.

For the kNN approach, we compare the performance by using only the query-document similarity for obtaining the relevance score (i.e. dropping the second term in Equation 1). On average this results in a drop of 5.6 percentage points, proving the effectiveness of injecting feedback documents in the kNN re-ranking approach.

For the CE experiments, we ablate if optimizing only the bias layers compared to fully fine-tuning the model affects the performance. We, therefore, repeat our query fine-tuning experiment but optimize all parameters of the model. On average, optimizing only the biases results in a 0.8% performance drop. However, the biases account only for 26k parameters, which is 0.11% of the entire model. This result is in line with other research showing that optimizing only a small subset of parameters results in comparable performance (Houlsby et al., 2019; Pfeiffer et al., 2020; Ben Zaken et al., 2022). This finding supports the query fine-tuning applicability from a memory perspective. While there might be many queries in a deployed system, and therefore many fine-tuned models, the required memory would not grow significantly. Furthermore, the memory requirements could be further reduced by only fine-tuning biases of certain components (Ben Zaken et al., 2022) or transformer layers (Rücklé et al., 2021).

	Robust	Covid	News	Touché	Avg.
BM25 without feedback documents					
	0.0459	0.1615	0.0551	0.1052	0.0919
kNN (Query Only)					
$k = 2$	0.3531	0.6611	0.2537	0.1637	0.3579
$k = 4$	0.3652	0.6486	0.2512	0.1649	0.3575
$k = 8$	0.3677	0.6854	0.2578	0.1687	0.3699
Avg.	0.3620	0.6650	0.2542	0.1658	0.3618
CE Query-FT (full)					
$k = 2$	0.4721	0.7168	0.3279	0.1797	0.4241
$k = 4$	0.5110	0.6872	0.3487	0.1858	0.4332
$k = 8$	0.5778	0.7644	0.3477	0.2021	0.4730
Avg.	0.5203	0.7228	0.3414	0.1892	0.4434
CE supervised + Query-FT (bias)					
$k = 2$	0.4540	0.7303	0.2716	0.2251	0.4203
$k = 4$	0.4896	0.7227	0.3657	0.2172	0.4488
$k = 8$	0.5353	0.7221	0.3390	0.2104	0.4517
Avg.	0.4930	0.7250	0.3254	0.2176	0.4403

Table 4: nDCG@20 results of ablations studies on the test set. The first experiment shows BM25 without using query expansion and removing the feedback documents from the evaluation. The next experiment ablates the performance of kNN by removing the influence of the relevant feedback documents. The third row shows results for fully fine-tuning the Cross-Encoder model, ablating fine-tuning only the bias layers. The last experiment ablates the MAML training, comparing it to standard supervised learning.

Finally, we investigate the impact of meta-learning by comparing it with supervised training. We follow the same setup as in MAML but replace meta-learning with standard supervised learning. We find that MAML training results in 0.5% improvement. We also note that supervised training is less stable than MAML. When increasing k, the performance intermittently drops (e.g. in TREC-Covid from $k = 2 \rightarrow 4$ and TREC-News from $k = 4 \rightarrow 8$), while MAML does not experience performance decreases.

6.4 Retrieval and Re-Ranking Latency

Results for the speed performance are reported in Figure 2. First, we note that performing query expansion does significantly increase retrieval speed. Depending on the number of feedback documents this is a 2.8 ($k = 2$) – 7.6 ($k = 8$) fold increase

[10] We have also experimented with combining BM25-QE, kNN and CE MAML + Query FT, however, have found it not to crucially outperform fusing only two rankings.

[11] See Appendix D for details on the distribution.

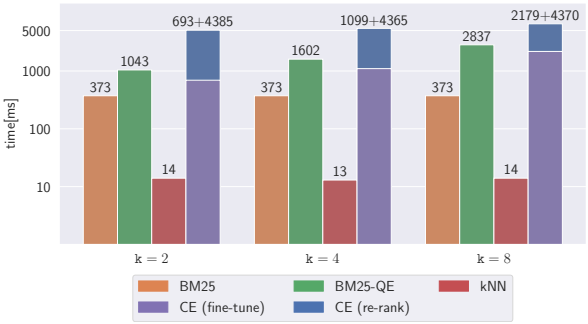

Figure 2: Average time in milliseconds (in log scale) for retrieval (BM25 and BM25-QE) and re-ranking (kNN and CE) 1000 documents. Average over all queries in the test sets. For the Cross-Encoder we separate the time for fine-tuning and re-ranking.

over BM25 without query expansion.[12]

For the re-ranking methods, we notice that the kNN approach is extremely fast. This is due to the fact that all heavy computations can be precomputed. This is promising since combining kNN and BM25-QE with rank fusion results in a 2.6% performance improvement over BM25-QE alone, while not significantly adding any latency. In contrast, the Cross-Encoder model takes the longest time. However, the time for fine-tuning the model is only a fraction of the total time ($\approx 22\%$ on average). The retrieval latency can generally be traded-off with the ranking performance by retrieving and re-ranking fewer documents.

7 Conclusion

In this work, we introduced a few-shot learning task for incorporating relevance feedback in neural re-ranking models. We further transformed existing IR datasets into the few-shot setting. Most importantly, we have introduced different methods for incorporating relevance feedback directly into neural re-ranking models. The proposed kNN approach is particularly computationally efficient, however, by itself, it cannot outperform BM25 with query expansion. Since the kNN method does not add significant latency to the re-ranking, it can be combined with BM25 query expansion, which outperforms the latter by 2.6% nDCG@20. Our second re-ranking method based on a Cross-Encoder model performs on par with BM25 with query expansion. Regarding its latency, we show that fine-tuning on a query basis is feasible since a majority of the time is spent on re-ranking and not fine-tuning. Similar to kNN, performing rank fusion of the two approaches yields a high performance gain of 5.2% nDCG@20. Advantageously, reciprocal rank fusion is very stable in our setting, mitigating weaknesses of individual model-task combinations.

8 Limitations

In this work, we investigate how relevance feedback can directly be incorporated into neural re-ranking models. While our best-performing approach improves the ranking performance by a large margin, it is inherently more computationally expensive compared with models that do not use any relevance feedback. We quantify this by reporting the latency of our approaches. The speed can be further reduced by re-ranking fewer documents, thereby trading off latency and performance. Further, we propose a kNN model that is computationally efficient and does not significantly add latency to query expansion models. Lastly, we recommend using our approach foremost in the information-seeking scenario, where users are particularly concerned about having accurate results rather than low latency.

For our methods, relevance feedback has to be explicitly collected from a user. While we believe in a information-seeking scenario users are more willing to provide explicit feedback, in this work, we did not explore using implicit or pseudo-relevance feedback. While this type of feedback is noisier, larger amounts are available.

In this work, we make use of simulated relevance feedback from existing relevance judgments. The re-ranked documents in the second stage will be biased toward the selected feedback documents. We leave to future work the integration of more diverse search results and investigation of position bias. However, we note that in preliminary experiments, we found that selecting random feedback documents from the first stage leads to worse performance.

To keep the degrees of freedom in our experiments reasonable and to facilitate evaluation, we do not experiment with an iterative relevance feedback setting. We instead focus on a single round of relevance feedback but vary the number of feedback documents. While related work has shown that iterative relevance feedback can further improve retrieval, there are diminishing gains with every round (Bi et al., 2019).

Our best-performing approach requires a train-

[12] For retrieval speed with varying number of expansion terms e see Appendix E.

ing dataset. Albeit small (depending on the task, the training dataset contains 22 - 90 queries), the model cannot be created without it. Since the model is targeted to a specific domain, we hypothesize that employing it on a different domain will result in worse performance than using the pre-trained model. To mitigate this, we also experiment with a model that does not use this intermediate fine-tuning step (CE Query-FT). Nevertheless, we encourage future work to look into combining unsupervised domain adaptation with our approaches to alleviate this limitation and potentially further improve performance.

Acknowledgements

We thank Max Glockner, Haritz Puerto, Rachneet Sachdeva, Gözde Gül Şahin, Thy Tran, and Kexin Wang for their insightful feedback on and reviews of earlier versions of the paper as well as the anonymous reviewers for their helpful comments and suggestions.

This work has been funded by the German Research Foundation (DFG) as part of the QASci-Inf project (grant GU 798/18-3) and the UKP-SQuARE project (grant GU 798/29-1). Furthermore, it has been supported by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 135–144. ACM.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. 2000. Bridging the lexical chasm: Statistical approaches to answer-finding. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, page 192–199, New York, NY, USA. Association for Computing Machinery.

Keping Bi, Qingyao Ai, and W. Bruce Croft. 2019. Iterative relevance feedback for answer passage retrieval with passage-level semantic match. In *Advances in Information Retrieval - 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14-18, 2019, Proceedings, Part I*, volume 11437 of *Lecture Notes in Computer Science*, pages 558–572. Springer.

Alexander Bondarenko, Lukas Gienapp, Maik Fröbe, Meriem Beloucif, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2021. Overview of touché 2021: Argument retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 12th International Conference of the CLEF Association, CLEF 2021, Virtual Event, September 21-24, 2021, Proceedings*, volume 12880 of *Lecture Notes in Computer Science*, pages 450–467. Springer.

Andrei Z. Broder. 2002. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10.

Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, page 659–666, New York, NY, USA. Association for Computing Machinery.

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*, SIGIR '09, page 758–759, New York, NY, USA. Association for Computing Machinery.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. 1987. The vocabulary problem in human-system communication. *Commun. ACM*, 30(11):964–971.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Christophe Van Gysel and Maarten de Rijke. 2018. Pytrec_eval: An extremely fast python interface to trec_eval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 873–876. ACM.

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

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 113–122, New York, NY, USA. Association for Computing Machinery.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Kalervo Järvelin and Jaana Kekäläinen. 2000. IR evaluation methods for retrieving highly relevant documents. In *SIGIR 2000: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 24-28, 2000, Athens, Greece*, pages 41–48. ACM.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 133–142, New York, NY, USA. Association for Computing Machinery.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. *ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT*, page 39–48. Association for Computing Machinery, New York, NY, USA.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur 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 Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Victor Lavrenko and W. Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 120–127, New York, NY, USA. Association for Computing Machinery.

Canjia Li, Yingfei Sun, Ben He, Le Wang, Kai Hui, Andrew Yates, Le Sun, and Jungang Xu. 2018. NPRF: A neural pseudo relevance feedback framework for ad-hoc information retrieval. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4482–4491, Brussels, Belgium. Association for Computational Linguistics.

Jimmy Lin. 2019. The simplest thing that can possibly work: Pseudo-relevance feedback using text classification. *CoRR*, abs/1904.08861.

Daniel Locke and Guido Zuccon. 2018. A test collection for evaluating legal case law search. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1261–1264, New York, NY, USA. Association for Computing Machinery.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Shahrzad Naseri, Jeffrey Dalton, Andrew Yates, and James Allan. 2021. Ceqe: Contextualized embeddings for query expansion. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part I*, page 467–482, Berlin, Heidelberg. Springer-Verlag.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. 1773.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y. Zhao, Yi Luan, Keith B. Hall, Ming-Wei Chang, and Yinfei Yang. 2021. Large dual encoders are generalizable retrievers. *CoRR*, abs/2112.07899.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Joseph Rocchio. 1971. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gerard Salton and Chris Buckley. 1990. Improving retrieval performance by relevance feedback. *Journal of the American society for information science*, 41(4):288–297.

Eric Schurman and Jake Brutlag. 2009. Performance related changes and their user impact. Presented at Velocity Web Performance and Operations Conference.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Ian Soboroff, Shudong Huang, and Donna Harman. 2018. TREC 2018 news track overview. In *Proceedings of the Twenty-Seventh Text REtrieval Conference, TREC 2018, Gaithersburg, Maryland, USA, November 14-16, 2018*, volume 500-331 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: Constructing a pandemic information retrieval test collection. *SIGIR Forum*, 54(1).

Ellen M. Voorhees. 2004. Overview of the TREC 2004 robust track. In *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004, Gaithersburg, Maryland, USA, November 16-19, 2004*, volume 500-261 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360, Seattle, United States. Association for Computational Linguistics.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Nancy Xin Ru Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020a. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020b. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Hao Wu and Hui Fang. 2013. An incremental approach to efficient pseudo-relevance feedback. In *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013*, pages 553–562. ACM.

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

HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 3592–3596, New York, NY, USA. Association for Computing Machinery.

ChengXiang Zhai and John D. Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management, Atlanta, Georgia, USA, November 5-10, 2001*, pages 403–410. ACM.

Xinyu Zhang, Xueguang Ma, Peng Shi, and Jimmy Lin. 2021. Mr. TyDi: A multi-lingual benchmark for dense retrieval. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 127–137, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: Contextualized Query Expansion for Document Re-ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4718–4728, Online. Association for Computational Linguistics.

A Dataset Details

Robust04 (Voorhees, 2004) is a dataset initially created to investigate the performance of poorly performing queries. Thereby, a collection with many judgments per query has been created and has since been used to test the robustness of IR models. We use the description field of the queries, which is a question or a single sentence of the search intent. The document collection contains news articles.

TREC-Covid (Voorhees et al., 2021) is an IR dataset in the biomedical domain consisting of questions about Coronavirus and scientific articles as document collection. It was collected in five iterative rounds. We use the question from the query set along with the documents from the COVID-19 Open Research Dataset (Wang et al., 2020a).[13] Documents are constructed by concatenating the title and abstract. Further, we remove exact duplicates from the feedback documents. In TREC-Covid, the documents are judged as relevant, partially relevant, or non-relevant. For the feedback documents, we consider only the relevant and non-relevant ones but include also partially relevant ones for evaluation.

TREC-News (Soboroff et al., 2018) is an Information Retrieval task based on a corpus provided by the Washington Post. We use the 2019 background linking task. In this setup, the goal is to find other relevant news articles that provide background information or further reading on a subject and help the user contextualize the current article. To have a concise query, we use the titles as query.

Webis-Touché 2020 (Bondarenko et al., 2021) is an argument retrieval dataset based on the args.me[14] corpus containing arguments scraped from debate websites.[15] Queries are formulated as questions. The dataset contains fine-grained annotations of documents on a scale from 0-7. We select documents with a relevance of at least 3 for our relevant feedback documents. Since Webis-Touché only contains very few non-relevant documents (i.e. relevance of 0), we augment them using BM25 negatives (Karpukhin et al., 2020), by selecting non-judged documents after rank 100 for the non-relevant feedback documents.

[13] We use the snapshot from 16-JULY-2020

[14] args.me/api-en.html

[15] debate.org, debatepedia.org, debatewise.org, idebate.org

B 1st Stage Retrieval

Table 5 shows BM25 retrieval performance with the relevance feedback documents removed ($F = ✗$) or included ($F = ✓$) in the retrieval results and evaluation.

	F	nDCG@20	R@100	R@1000
Robust	✓	0.3292	0.2256	0.5021
	✗	0.0459	0.1488	0.4464
Covid	✓	0.5597	0.0963	0.3686
	✗	0.1615	0.0783	0.3556
News	✓	0.2993	0.4039	0.7406
	✗	0.0551	0.3137	0.7017
Touché	✓	0.5134	0.4067	0.6461
	✗	0.1052	0.2887	0.5717
Avg.	✓	0.4254	0.2831	0.5644
	✗	0.0919	0.2074	0.5188

Table 5: Retrieval results using BM25 on the test set with the query only. For F = ✗ the feedback documents have been removed from the retrieved documents and the ground truth for computing the evaluation metrics.

C Zero-Shot Baselines

	Robust	Covid	News	Touche	Avg.
DPR-single (109M) (Karpukhin et al., 2020)					
$k = 2$	0.1335	0.4232	0.1913	0.0510	0.1997
$k = 4$	0.1254	0.4359	0.1922	0.0547	0.2020
$k = 8$	0.1510	0.4749	0.1981	0.0603	0.2211
Avg.	0.1366	0.4447	0.1938	0.0553	0.2076
DPR-multi (109M) (Karpukhin et al., 2020)					
$k = 2$	0.2570	0.4466	0.2191	0.0917	0.2536
$k = 4$	0.2662	0.4312	0.2156	0.0924	0.2513
$k = 8$	0.2690	0.4431	0.2188	0.0959	0.2567
Avg.	0.2641	0.4403	0.2178	0.0933	0.2539
ANCE (125M) (Xiong et al., 2021)					
$k = 2$	0.3481	0.6671	0.3061	0.1511	0.3681
$k = 4$	0.3574	0.6825	0.3064	0.1497	0.3740
$k = 8$	0.3662	0.6834	0.3137	0.1544	0.3794
Avg.	0.3572	0.6777	0.3087	0.1517	0.3738
MiniLM (23M) (Wang et al., 2020b)					
$k = 2$	0.3824	0.5795	0.2800	0.1433	0.3463
$k = 4$	0.3843	0.5847	0.2831	0.1570	0.3523
$k = 8$	0.4044	0.6265	0.2856	0.1668	0.3708
Avg.	0.3903	0.5969	0.2829	0.1557	0.3565
TAS-B (66M) (Hofstätter et al., 2021)					
$k = 2$	0.3637	0.6475	0.2549	0.1397	0.3515
$k = 4$	0.3666	0.6568	0.2612	0.1570	0.3604
$k = 8$	0.3736	0.6607	0.2576	0.1553	0.3618
Avg.	0.3680	0.6550	0.2579	0.1507	0.3579
GPL (66M) (Wang et al., 2022)					
$k = 2$	0.3741	0.6626	0.3049	0.1588	0.3751
$k = 4$	0.3810	0.6781	0.3090	0.1646	0.3832
$k = 8$	0.3880	0.6810	0.3140	0.1722	0.3888
Avg.	0.3810	0.6739	0.3093	0.1652	0.3824
GTR-base (110M) (Ni et al., 2021)					
$k = 2$	0.3986	0.6157	0.2930	0.1647	0.3680
$k = 4$	0.3999	0.6436	0.3065	0.1751	0.3813
$k = 8$	0.4137	0.6203	0.3096	0.1828	0.3816
Avg.	0.4041	0.6266	0.3031	0.1742	0.3770
GTR-large (335M) (Ni et al., 2021)					
$k = 2$	0.4224	0.6194	0.3405	0.1727	0.3887
$k = 4$	0.4264	0.6494	0.3652	0.1756	0.4041
$k = 8$	0.4367	0.6416	0.3693	0.1801	0.4069
Avg.	0.4285	0.6368	0.3583	0.1761	0.3999
GTR-XL (1.24B) (Ni et al., 2021)					
$k = 2$	0.4256	0.6258	0.3852	0.1688	0.4013
$k = 4$	0.4283	0.6459	0.3902	0.1765	0.4102
$k = 8$	0.4357	0.6530	0.3902	0.1813	0.4151
Avg.	0.4299	0.6416	0.3885	0.1755	0.4089

(a) nDCG@20 on the validation set.

	Robust	Covid	News	Touche	Avg.
DPR-single					
$k = 2$	0.1064	0.4690	0.1738	0.0824	0.2079
$k = 4$	0.1144	0.4822	0.1911	0.0815	0.2173
$k = 8$	0.1230	0.5016	0.1930	0.0849	0.2256
Avg.	0.1146	0.4843	0.1859	0.0830	0.2169
DPR-multi					
$k = 2$	0.2234	0.4622	0.2058	0.1190	0.2526
$k = 4$	0.2372	0.4558	0.2122	0.1199	0.2563
$k = 8$	0.2447	0.4846	0.2160	0.1249	0.2675
Avg.	0.2351	0.4675	0.2113	0.1213	0.2588
ANCE					
$k = 2$	0.3523	0.6826	0.3218	0.1744	0.3828
$k = 4$	0.3561	0.6887	0.3087	0.1786	0.3830
$k = 8$	0.3613	0.7023	0.3135	0.1766	0.3884
Avg.	0.3566	0.6912	0.3147	0.1765	0.3847
MiniLM					
$k = 2$	0.3531	0.6611	0.2537	0.1637	0.3579
$k = 4$	0.3652	0.6486	0.2512	0.1649	0.3575
$k = 8$	0.3677	0.6854	0.2578	0.1687	0.3699
Avg.	0.3620	0.6650	0.2542	0.1658	0.3618
TAS-B					
$k = 2$	0.3658	0.6872	0.2813	0.1486	0.3707
$k = 4$	0.3833	0.6845	0.2830	0.1567	0.3769
$k = 8$	0.3952	0.6878	0.2787	0.1557	0.3793
Avg.	0.3814	0.6865	0.2810	0.1537	0.3756
GPL					
$k = 2$	0.3623	0.6979	0.3011	0.1617	0.3808
$k = 4$	0.3800	0.7007	0.3075	0.1663	0.3886
$k = 8$	0.3873	0.7129	0.3191	0.1681	0.3969
Avg.	0.3766	0.7038	0.3092	0.1654	0.3888
GTR-base					
$k = 2$	0.3638	0.7051	0.3177	0.1988	0.3963
$k = 4$	0.3799	0.7188	0.3262	0.2080	0.4082
$k = 8$	0.3875	0.7222	0.3250	0.2130	0.4119
Avg.	0.3771	0.7154	0.3230	0.2066	0.4055
GTR-large					
$k = 2$	0.4003	0.6844	0.3308	0.1941	0.4024
$k = 4$	0.4070	0.6851	0.3308	0.1984	0.4053
$k = 8$	0.4093	0.6850	0.3426	0.2012	0.4095
Avg.	0.4056	0.6848	0.3347	0.1979	0.4057
GTR-XL					
$k = 2$	0.4024	0.6804	0.3456	0.2054	0.4084
$k = 4$	0.4135	0.6858	0.3387	0.2130	0.4127
$k = 8$	0.4157	0.6764	0.3441	0.2140	0.4125
Avg.	0.4105	0.6808	0.3428	0.2108	0.4112

(b) nDCG@20 on the test set.

Table 6: Zero-shot nDCG@20 results on the residual collection. Model sizes are reported in parentheses. For DPR, MiniLM, TAS-B and GTR we use the checkpoints from the sentence-transformers library (Reimers and Gurevych, 2019). For GPL, we use the self-miner model.

D BM25-QE and Neural Re-Ranker Top 20 Analysis

Figure 3 presents the number of documents that BM25-QE and the respective neural re-ranking method have in common in the top 20 retrieval results. It shows that while there is some overlap, the methods rank different documents on top.

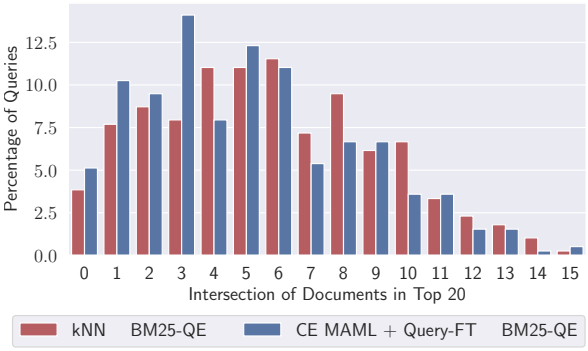

Figure 3: Overlap of documents in the top 20 between BM25-QE and two neural re-ranking methods.

E Retrieval Speed with Query Expansion

Figure 4 presents the average time duration per query when varying the number of expansion terms when using BM25 with query expansion.

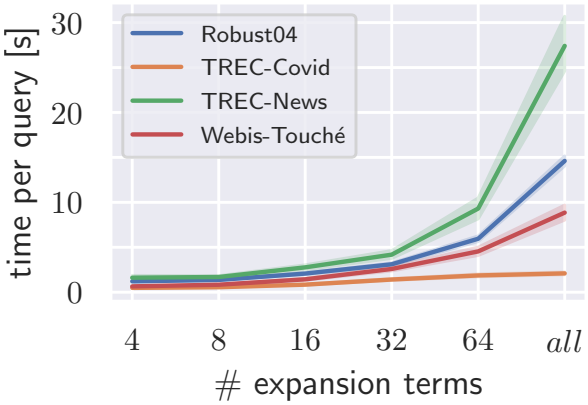

Figure 4: Average duration per query with varying the number of expansion terms.

F Results Validation Set

	Robust	Covid	News	Touche	Avg.
BM25-QE					
$k = 2$	0.4135	0.6340	0.4195	0.2113	0.4196
$k = 4$	0.4442	0.6103	0.4346	0.2255	0.4287
$k = 8$	0.4870	0.6894	0.4798	0.2466	0.4757
Avg.	0.4483	0.6446	0.4446	0.2278	0.4413
kNN					
$k = 2$	0.4346	0.6159	0.3702	0.1301	0.3877
$k = 4$	0.4267	0.6420	0.3955	0.1523	0.4041
$k = 8$	0.4715	0.6771	0.4475	0.1742	0.4426
Avg.	0.4443	0.6450	0.4044	0.1522	0.4115
CE Zero-Shot					
$k = 2$	0.3902	0.6598	0.2917	0.1618	0.3759
$k = 4$	0.3919	0.6786	0.3249	0.1700	0.3914
$k = 8$	0.4064	0.6713	0.3416	0.1799	0.3998
Avg.	0.3962	0.6699	0.3194	0.1706	0.3890
CE Query-FT					
$k = 2$	0.4511	0.6624	0.3179	0.1786	0.4025
$k = 4$	0.4676	0.7381	0.3870	0.1913	0.4460
$k = 8$	0.5214	0.7551	0.4130	0.2041	0.4734
Avg.	0.4800	0.7186	0.3726	0.1913	0.4406
CE MAML + Query FT					
$k = 2$	0.4640	0.6691	0.3276	0.2186	0.4198
$k = 4$	0.5096	0.7595	0.3845	0.2216	0.4688
$k = 8$	0.5361	0.7729	0.4035	0.2237	0.4840
Avg.	0.5032	0.7339	0.3719	0.2213	0.4576
Rank Fusion: kNN & BM25-QE					
$k = 2$	0.4551	0.6992	**0.4273**	0.1931	0.4437
$k = 4$	0.4661	0.6877	**0.4574**	0.2144	0.4564
$k = 8$	0.5155	0.7261	**0.5039**	0.2243	0.4924
Avg.	0.4789	0.7043	**0.4629**	0.2106	0.4642
Rank Fusion: CE MAML + Query-FT & BM25-QE					
$k = 2$	**0.4911**	**0.7156**	0.4190	0.2389	**0.4661**
$k = 4$	**0.5359**	**0.7610**	0.4516	**0.2468**	**0.4988**
$k = 8$	**0.5724**	**0.7998**	0.4708	**0.2475**	**0.5226**
Avg.	**0.5331**	**0.7588**	0.4471	**0.2444**	**0.4958**

Table 7: nDCG@20 results on the validation set. The top-performing result is shown in boldface, runner-up is underlined.

G Ablations Validation Set

	Robust	Covid	News	Touche	Avg.
BM25 without feedback documents					
	0.0451	0.1802	0.0403	0.0861	0.0879
kNN (Query Only)					
$k = 2$	0.3824	0.5801	0.2800	0.1433	0.3464
$k = 4$	0.3843	0.5847	0.2831	0.1570	0.3523
$k = 8$	0.4044	0.6265	0.2856	0.1668	0.3708
Avg.	0.3903	0.5971	0.2829	0.1557	0.3565
CE Query-FT (full)					
$k = 2$	0.4876	0.6808	0.3397	0.2005	0.4272
$k = 4$	0.5003	0.7700	0.3896	0.2129	0.4682
$k = 8$	0.5598	0.7802	0.4180	0.2074	0.4914
Avg.	0.5159	0.7437	0.3824	0.2069	0.4622
CE supervised + Query-FT (bias)					
$k = 2$	0.4579	0.7260	0.3189	0.2131	0.4290
$k = 4$	0.4876	0.7286	0.3854	0.2256	0.4568
$k = 8$	0.5200	0.7449	0.4241	0.2296	0.4797
Avg.	0.4885	0.7332	0.3761	0.2228	0.4551

Table 8: nDCG@20 results on the validation set of the ablation studies.

H Models, Hyperparameters & Hardware

Hardware & Settings for Latency Experiments	
Elasticsearch Version	7.11.2
Elasticsearch Settings	Single Shard, Cache cleared after each query
Elasticsearch Hardware	24 Intel Xeon CPU E5-2620 v2 @ 2.10GHz
GPU (for kNN and CE)	NVIDIA P100, 16GB
Models	
kNN	sentence-transformers/all-MiniLM-L6-v2
Cross-Encoder	cross-encoder/ms-marco-MiniLM-L-6-v2
Model Parameters	
kNN	22.7M
Cross-Encoder	22.7M
Cross-Encoder (biases only)	26k
Training Settings	
Optimizer	AdamW (Loshchilov and Hutter, 2019)
Optimizer (MAML)	SGD
Hyperparameters	
Learning rates for Query-FT, MAML and supervised training	$\{2 \times 10^{-3}, 2 \times 10^{-4}, 2 \times 10^{-5}\}$
Epochs query fine-tuning	$1 - 8$
Evaluation Libraries	
PYTREC-EVAL Version	0.5

Table 9: Hyperparameters and models used in our experiments. The best learning rate and the number of epochs have been selected according to the nDCG@20 validation performance.

I 2nd Stage Retrieval: BM25 with Query Expansion

e	k = 2	k = 4	k = 8	Avg.
4	0.6164	0.6242	0.6520	0.6309
8	0.6167	0.6369	0.6686	0.6407
16	**0.6266**	**0.6470**	**0.6700**	**0.6479**
32	0.6122	0.6262	0.6463	0.6283
64	0.5704	0.5776	0.5868	0.5783
all	0.5722	0.5828	0.5823	0.5791

(a) Recall@1000 on the validation set.

e	$k = 2$	$k = 4$	$k = 8$	Avg.
4	0.6187	0.6300	0.6566	0.6351
8	**0.6280**	**0.6414**	0.6721	**0.6472**
16	0.6195	0.6400	**0.6736**	0.6444
32	0.6039	0.6209	0.6477	0.6242
64	0.5597	0.5729	0.5843	0.5723
all	0.5723	0.5771	0.5828	0.5774

(b) Recall@1000 on the test set.

Table 10: Recall@1000 results for BM25 with query expansion on the validation (a) and test (b) set for varying number of expansion terms e extracted per document. In bold best performing setting and in underline runner-up. The last column shows the average over all k. Although using $e = 8$ performs best on the test set, we conduct subsequent experiments with $e = 16$ since we also tune other hyperparameters on the validation set.