Evaluating the Suitability of Interpolation-Based Re-ranking For Ad-Hoc Retrieval

Lucia Navarčíková
Supervisor(s): Avishek Anand, Jurek Leonhardt
[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student:Lucia Navarčíková
Final project course: CSE3000 Research Project
Thesis committee: Avishek Anand, Jurek Leonhardt, Alan Hanjalic

Abstract

Interpolation-based re-ranking emerged to make dense retrieval possible in low-latency applications such as web engine search. However, to this day there is no clear winner among the different ranking approaches. Moreover, missing document scores in hybrid retrieval have not been investigated in detail. This paper aims to address this by comparing the interpolation-based re-ranking with dense and hybrid retrieval approaches in terms of ranking performance and latency. It goes to show that while interpolation-based re-ranking has a notable latency advantage, hybrid retrieval achieves best performance in ranking metrics for an in-domain setting. Additionally, differences in missing document score techniques are slight with zero imputation emerging on top.

1 Introduction

With the estimate of 181 zettabytes[1] of information available online by 2025 [1], identifying relevant websites is crucial. The task referred to as *ad-hoc retrieval* consists of a user defining a query in natural language, based on which documents are retrieved in the order of their relevance [2]. The relevance of a document is indicated by a score, representing the similarity of the query to the document. Documents are ranked according to their score and returned to the user with a document of the highest score in the first place.

Research in this domain has been on *neural rankers*, which are typically built on pre-trained language models such as BERT [3]. Allowing bidirectional training enhances the understanding of underlying semantic principles, making them less susceptible to the *vocabulary mismatch problem*. Vocabulary mismatch problem occurs when the language utilized in a query and document deviates, but the document still contains information relevant to the user. *Sparse* retrieval approaches, based on lexical matching, will not retrieve the document, but the ability of neural rankers to consider context can overcome this issue. However, with the increasing complexity of these new models comes increased latency, which is unsuitable for real-time web search engines especially when working with large corpora.

The FAST-FORWARD indexes framework proposed by Leonhardt et al. [4] addresses this issue by utilizing separate encoders for query and document along with *interpolation-based re-ranking*. It maintains low-latency despite using a semantic re-ranker by using pre-computed document representations. Interpolation-based re-ranking employs a lexical model for the first retrieval step, therefore limiting the number of documents to be processed by the semantic model in the re-ranking step. This leads to drastic improvements in efficiency for large corpora, where computing semantic scores for the whole corpus would be infeasible. The results provided by Leonhardt et al. [4] focus mostly on MS MARCO [5]. The MS MARCO dataset is often utilized for model training due to its large size, however, it is imperative to consider additional datasets from different domains to ensure diversity in evaluating benchmarks, as acknowledged by Thakur et al. [6].

An alternative to interpolation-based re-ranking is *hybrid retrieval*. Hybrid retrieval employs a lexical and a semantic model to retrieve documents in parallel and their score interpolation forms the final score. Consequently, this leads to cases where both models do not retrieve the identical set of documents, leading to *missing document scores*. Hybrid retrieval models diverge in addressing this issue with little to no discussion on the effect of different approaches in the same setting. The paper aims to address this gap by implementing various missing score approaches.

[1] 10^{21} bytes

The variety of the approaches described can be attributed to the fact that there is no clear winner, no single approach can consistently outperform others on all datasets and predefined metrics [6].

Henceforth, this paper aims to answer the following research questions:
RQ1: How does interpolation-based re-ranking (using FAST-FORWARD indexes) compare to dense and hybrid retrieval approach in terms of ranking performance and latency?
RQ2: What is the importance of the lexical component in hybrid retrieval models and interpolation-based re-ranking (implemented using FAST-FORWARD indexes), respectively?
RQ3: To what extent do missing document scores impact ranking performance in hybrid retrieval models and how can this problem be mitigated?

We find that interpolation-based re-ranking outperforms both dense and hybrid retrieval in latency and achieves best ranking performance in out-of-domain datasets. Furthermore, hybrid retrieval improves recall, but doubles the per-query latency compared to interpolation-based re-ranking. Lastly, we find that imputing zero in case of a missing score leads to best ranking results for hybrid retrieval.

2 Background

This section delves into the details of paper background. It defines sparse retrieval and its limitations, followed by examining the motivation behind dense neural rankers and related architecture choices. It continues by giving an overview of hybrid retrieval models and the techniques for dealing with the missing document scores. Finally, it introduces retrieve-and-re-rank with its association to interpolation-based re-ranking and FAST-FORWARD indexes.

2.1 Sparse Retrieval

Sparse approaches such as BM25 [7] have been the default approach in information retrieval. They assume sparse document representations and use lexical similarity to determine the ranking of a document. The probabilistic approach utilizes term frequency-inverse document frequency (TF-IDF) weighting and document length normalization. Inverse term-frequency is calculated as a logarithmic function, thereby diminishing weights of most common words. It assigns higher term weights to rare terms in the corpus, while document length normalization ensures large documents do not dominate ranking based on their size alone. The implementation of BM25 and its extensions [8], [9] is limited to exact term matching causing the *vocabulary mismatch problem.* However, even with new developments sparse retrieval remains relevant especially in settings with low computational resources, where BM25 [7] remains a strong baseline [6].

Aside from traditional term-weighing approaches, there has been an effort to utilize neural networks in sparse retrieval, following numerous breakthroughs in dense retrieval. DeepCT [10] employs supervised learning of term weights from BERT [3] representations, which contain semantic information. By keeping the inverted index structure based on term frequency, but nevertheless incorporating context information, it is able to achieve large improvements in first-stage retrieval accuracy [10]. However, this approach does not address the vocabulary mismatch problem, thereby DeepCT additionally incorporates query expansion. Other first-stage rankers in this category include SPLADE [11], which predicts term importance from BERT WordPiece vocabulary with query and document expansion, or

UniCoil [12], which leverages BERT for feature vectors and applies regularization to further reduce vector dimensionality.

2.2 Dense Retrieval

Compared to sparse approaches, dense models work with more complex document representations consisting of lower-dimensional dense vectors. This allows for document similarity to be expressed as distance in n-dimensional vector space. The nature of these embeddings enables the utilization of contextual information, rather than relying on exact term matching. With the rising popularity of Large Language Models, dense neural rankers models have been built on top the *transformer* architecture [13].

Transformer architecture was developed in efforts to allow for increased parallelization compared to sequential computing needed in convolutional and recurrent neural network approaches. By eliminating sequence-based RNN or convolution and relying entirely on self-attention mechanisms, it can make use of different positions relative to a single sequence in constant number of operations [13]. Furthermore, it allows for making global dependencies between input and output [13], thereby making it more context-aware.

BERT [3], which stands for Bidirectional Encoder Representations from Transformers, extended this approach by adding bidirectional representation by joining both left and right context in all layers. Previous approaches, such as GPT by OpenAI [14], used left-to-right transformer architecture, which Devlin et al. [3] argue leads to suboptimal performance. BERT utilizes Masked-Language-Model pre-training objective, where some tokens of the input are randomly masked and the model is trained to infer them from the context. This allows for understanding the broader context and state-of-the-art results with minimal effort [3]. The promising results led to an increase of BERT-based dense models in information retrieval [15]. An illustration of this is ColBERT [16], which encodes the document and query using BERT [3] separately and then employs a late interaction step for their similarity.

Additionally, dense retrieval models can be divided into two groups: *cross-attention* and *dual-encoders*. Cross-attention models take as an input a concatenation of the document and a query. By performing early-stage fusion of the input embeddings, cross-attention models are well suited for problems involving comparisons between paired textual inputs such as question answering task [17]. However, they are unsuited for real-time applications as increased input size results in higher query processing time. Dual encoders circumvent this issue by using a separate encoder for both query and the document. Consequently, the document representation can be pre-computed in the indexing stage, and retrieval is performed as approximate nearest neighbour search in the embedding space given an encoded query. Moreover, queries tend to be shorter than documents, as shown on average number of words in MS MARCO, where queries have on average 5.96 words with document average being 55.98 words as reported by Thaker et al. [6]. As a result, dual-encoder architecture is more suited for ad-hoc retrieval with low latency constraints.

2.3 Hybrid Retrieval

Hybrid retrieval models utilize both a sparse and a dense model in parallel for retrieval, and then combine the scores to determine the final ranking. As both models retrieve documents independently, there are cases when the sets of documents retrieved by two models are not identical, leading to a missing score for interpolation purposes. Hybrid implementation models fail to achieve consensus on this topic and the issue is dealt with on a model-to-model basis. Popular hybrid models include CLEAR [18], which applies BM25 [7] as a lexical retriever with Siamese framework based on BERT [15] in a single-stage multi-retrieval approach. By training the dense model using a residual method and not independent retrieval, the dense model "corrects" mistakes made by BM25 by supplementing semantic information. Another hybrid approach, COIL [19] produces document representation tokens with a dense approach and stores them in the inverted index, relying on vector similarity of document token representations for retrieval.

2.4 Interpolation-based Re-ranking

Interpolation-based re-ranking was designed to decrease the number of documents to be scored by the dense retriever as computing document scores for the whole corpus is infeasible for real-time applications. In the first stage, a sparse model is used to retrieve the documents and subsequently the more computationally expensive dense model is employed to re-rank them. In the retrieval step a sparse model assigns a score to each document, determining its initial ranking. Consequently the top-k relevant candidates are passed on to the dense model for re-ranking. The sparse score can be further utilized by interpolating the scores of dense and sparse models to determine the final ranking. Therefore given a query q and a document d the final score can be computed as

$$\phi(q,d) = \alpha \cdot \phi_S(q,d) + (1-\alpha) \cdot \phi_D(q,d) \tag{1}$$

with $\phi_S(q,d)$ and $\phi_D(q,d)$ being the scores of sparse and dense model respectively. Parameter α in range [0,1] determines the weight of the sparse score. The ranking performance of this approach relies heavily on the number of candidates selected, so called *retrieval depth*, as well as on the choice of the chosen sparse model. This approach decreases end-to-end latency by limiting the documents to be processed by a dense ranker.

FAST-FORWARD indexes [4] implement interpolation-based re-ranking, with additional support for *sequential coalescing* to reduce the size of the final index. One of the key differentiators of this framework lies in the separate usage of query and document encoders during the re-ranking step, inspired by Jung et al. [20]. This stems from the assumption that queries tend to be shorter in nature than the documents and can therefore be encoded differently.

3 Experimental Setup

This section will cover the experiments conducted in this study and the relevant implementation details. It first goes to describe the different models and the retrieval approaches utilized for this purpose. Afterwards, there are arguments for the suitability of the datasets used in the experiment, disclosing their relevant attributes. Lastly, it delves into query latency experiment structure.

3.1 Retrieval Approaches Implementation Details

To be able to reliably evaluate the suitability of interpolation-based re-ranking for ad-hoc retrieval, the following approaches were considered:

3.1.1 Retrieval Models

BM25 [7] is a sparse retrieval model using the inverted index structure. The approach was developed as an extension of TF-IDF [21], which considers the frequency of occurrence of the term in the document as well as across the whole corpus. Inverse Document Frequency (IDF) of can be expressed as:

$$idf(q_i) = \log \frac{N - df(q_i) + 0.5}{df(q_i) + 0.5} \tag{2}$$

where N is the total number of documents in the corpus and $df(q_i)$ is the number of documents containing term q_i.

Given query q and document d, the BM25 score is computed as a sum of term weights for every term q_i in document d:

$$BM25(q,d) = \sum_{q_i:tf(q_i,d)>0} \frac{idf(q_i) \cdot tf(q_i,d) \cdot (k_1 + 1)}{tf(q_i,d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \cdot \frac{k_3 \cdot qtf(q_i,q)}{k_3 + qtf(q_i,q)} \tag{3}$$

where $qtf(q_i,q)$ is query frequency, i.e. number of times q_i occurs in document d and $f(q_i,d)$ is the frequency of term q_i in document d. Additionally, $\frac{|d|}{avgdl}$ is the length of document d divided by the average document length in the collection. By normalizing the length of the document, longer documents won't dominate the top ranging sports purely due to larger word count. The parameters of BM25 k_1, b and k_3 are set to 1.2d, 0.75d and 8d respectively. In all experiments the used implementation is from the PyTerrier library [22].

Second model in this study, **TCT-ColBERT** [23], is a model created by using knowledge distillation on the late-interaction ColBERT [16] model. Late-interaction in ColBERT is achieved by encoding query q and document d into two separate sets of contextual embeddings. Instead of interaction within and across the document and query at the same time such as in BERT [3], ColBERT delays this interaction, therefore reducing the computational complexity during runtime. At the last step, each query embedding interacts with all document embeddings using a so called MaxSim operator. The MaxSim operator computes the maximum similarity (e.g. cosine similarity), which is then summed across all query terms into a single scalar. Given the query q and document d, the MaxSim operator can be expressed as:

$$\phi_{MaxSim}(\mathbf{q}, \mathbf{d}) = \sum_{i \in |E_q|} \max_{j \in |E_d|} \langle \eta_q (E_{q_i}), \eta_d (E_{dj}) \rangle, \tag{4}$$

where η is composed of:

$$\begin{aligned} \eta_q(\mathbf{x}) &= Normalize(Conv1D(\mathbf{x})) \\ \eta_d(\mathbf{x}) &= Filter(Normalize(Conv1D(\mathbf{x})) \end{aligned} \tag{5}$$

and E_q and E_d are query and document embedding respectively. Despite Khattab et al. [16] claiming ColBERT can be used to retrieve top-k results from large corpora instead of merely re-ranking output of sparse retrieval models, Lin et al. [23] argue that calculating MaxSim over the whole document collection is infeasible. Instead they propose TCT-ColBERT [23], which applies knowledge distillation in the form of a tightly coupled teacher and student model. The teacher is ColBERT with the student model starting as a Siamese network with

BERT-based bi-encoders using average pooled embeddings instead of MaxSim. Formally given a query q and document d, PoolDot can be defined as:

$$\phi_{PoolDot}(\mathbf{q}, \mathbf{d}) = \langle Pool\left(E_q\right), Pool\left(E_d\right)\rangle \tag{6}$$

where the Pool operator can be the average or the maximum pooling over the token embeddings (TCT-ColBERT implements the former). The training of TCT-ColBERT consists of 2 stages: fine-tuning the teacher utilizing MaxSim, and distilling its knowledge into the student model utilizing pooled embeddings. The loss function for training is composed of the predicted passage relevance from the teacher (ColBERT) using softmax cross entropy, as well as Kullback-Leibler divergence between sampled the probability distributions of the teacher and student models. As a consequence TCT-ColBERT simplifies relevance computation into dot product over the pooled query encodings during runtime, which allows for single step Approximate Nearest Neighbour (ANN) search.

3.1.2 Retrieval Approaches

Interpolation-based re-ranking is implemented using FAST-FORWARD indexes [4]. The framework facilitates a multi-stage retrieval pipeline, consisting of a lexical retriever and a dense model for re-ranking with linear interpolation of the scores. The framework employs dual-encoder architecture allowing offline document indexing and reducing computation constraints during runtime, leading to fast query processing due to constant look up of dense vector representations [4].

Large documents are split into passages, where the score of a document is calculated as the maximum passage score to reduce processing time. Formally this is expressed as:

$$\phi_D(q, d) = \max_{p_i \in d} \phi_D\left(q, p_i\right) \tag{7}$$

with p_i being the i-th passage of document d. Additionally, for processing larger documents, FAST-FORWARD indexes employ *sequential coalescing*, where similar adjacent passages combine their vector representations. This allows for smaller index size as well as smaller query processing time [4].

In this paper, the full pipeline consists of first stage retrieval of 1000 documents per query by BM25[7] followed by TCT-ColBERT[23] as a re-ranker, all implemented in in PyTerrier. The interpolation parameter α will be determined based on the ranking performance on a test set for each dataset in respect to $nDCG\_10$ from the range [0.05, 0.1, 0.25, 0.5, 0.75, 0.9].

Dense retrieval consists of a single-retrieval stage, where documents are retrieved from the whole corpus using a dense model, whose scores determine the final ranking. In this paper dense retrieval is implemented using Pyserini [24], consisting of retrieval of 1000 documents per query by TCT-ColBERT [23]. The implementation uses flat FAISS index with the `castorini/ tct_colbert-msmarco` query encoder. More information on the TCT-ColBERT architecture can be found in section 3.1.1

Hybrid Retrieval is implemented using Pyserini [24], consisting of retrieval of 1000 documents by both BM25 [7] and TCT-ColBERT [23] in parallel. The final ranking is determined by linear interpolation of the lexical and the semantic score. To account for possible differences in scale in ranking scores for these two models, in addition to using original scores, normalized scores are utilized (implemented using min-max scaling). Aside from α parameter tuning for interpolation, multiple approaches for missing score techniques are investigated, namely: dropping the document from the ranking completely or setting the missing score to zero, the average score and the median score. The missing score techniques are evaluated on the test set for each dataset.

3.2 Datasets

The ranking performance of the retrieval approaches is evaluated on the following datasets:

FiQA-2018 [25] is a dataset from a financial domain for question answering task, where documents consist of microblogs, news articles or reports. To make the datasets more diverse in their domain, NF Corpus was chosen. **NF Corpus** [26] is a dataset from the biomedical domain focused on biomedical information retrieval. The queries are formulated in layman's English and its corpus is composed of scientific articles from www.NutritionFacts.org . As both of the previous datasets are small in size, MS MARCO is included to provide a large data perspective. **MS MARCO** [5] is a large scale dataset consisting of anonymized questions from Bing and a human generated answer. The small version of the development set is used for hyperparameter tuning with the TREC-DL-Psg'19 utilized for final evaluation. It is a widely used dataset for bench marking state-of-the-art information retrieval models. The overview of the different datasets can be found in Table 1.

Dataset Name	**Task**	**Domain**	**Corpus**	**Query**
FiQA-2018	Question Answering	Finance	57638	6648
NF Corpus	Information Retrieval	Bio-Medical	3633	323
MS MARCO	Passage-Retrieval	Misc	8841823	6980

Table 1: Dataset Specifications. The number of documents is indicated along with the number of queries available

3.3 Evaluation Metrics

Recall (R) is an evaluation metric defining the fraction of relevant documents in the ranking. Relevance of a document d for a query q is denoted by $Rel(q,d)$ and can have either value 1 or 0 (in case of binary relevance). We define the ideal ranking as R_i and ranking obtained after retrieval as R_o. Therefore given a query q and rankings R_i and R_o, recall is defined as:

$$R = \frac{\sum_{d \in R_o} Rel(d,q)}{\sum_{d' \in R_i} Rel(d',q)} \tag{8}$$

In many cases it is desirable to calculate recall over the top k candidates, which can be achieved by setting a cut-off threshold. In this paper the cut-off is 100, with the metric denoted as R_{100} .

(Mean) Reciprocal Rank (RR) is a precision-focused metric that takes into account the obtained ranking of the first relevant document. For query set Q, mean reciprocal rank can be calculated as a multiplicative inverse of the rank of the first relevant document in the obtained ranking R_o for all queries. Formally, given a query set Q and a ranking R_o, mean reciprocal rank can be defined as:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{9}$$

where $rank_i$ refers to the rank of the first relevant document in the ranking R_o for query q_i. Similar to R_{100}, RR can have a cut-off value to consider only top k retrieved documents. In this paper we consider RR_{10}.

Normalized Discounted Cumulative Gain (nDCG) is a ranking quality metric [27]. It compares the obtained ranking R_o to the ideal ranking R_i where all relevant documents are at the top of the list . For each query the cumulative gain is calculated as the sum of the relevance for the top k documents. Discounted cumulative gain takes into account the rank of the document, such that relevance of higher ranks influences the resulting score more. In cases where there is a disparity between the number of relevant documents for each query, normalization is added for fair comparison. The normalization is achieved by dividing the obtained DCG by Ideal Discounted Cumulative Gain (IDCG) - the discounted cumulative gain of the best possible ranking (R_i). Formally, given a query q, ranking R_o composed of documents d_1,....d_n and ranking R_i composed of documents d'_1,....d'_n, normalized discounted cumulative gain can be expressed as:

$$NDCG = \frac{DCG(R_o)}{DCG(R_i)} = \frac{\sum_{i=1}^{n(R_o)} \frac{Rel(q,d_i)}{\log_2(i+1)}}{\sum_{i=1}^{n(R_i)} \frac{Rel(q,d'_i)}{\log_2(i+1)}} \tag{10}$$

where n is the number of documents. The metric has a cut-off value, determining the top k documents considered for the comparison. In this paper we report $NDCG_{10}$.

3.4 Latency Measurements

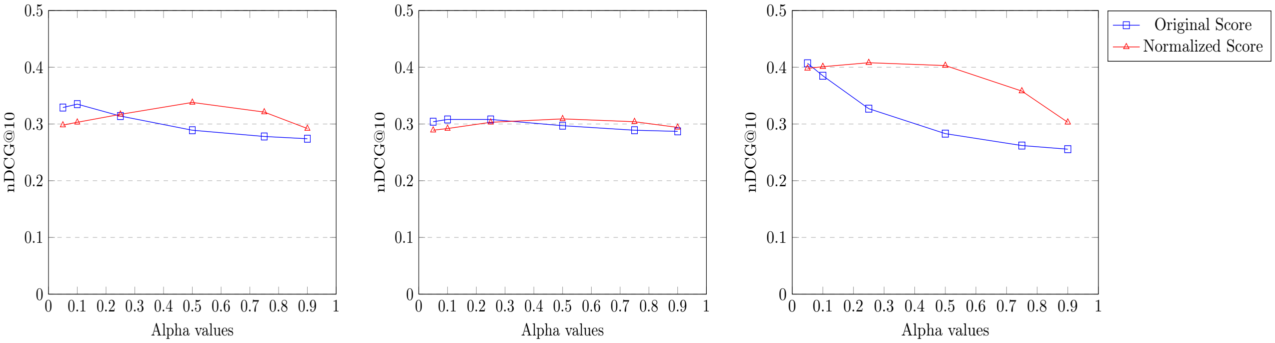
To determine which approach is best suited for search engines, i.e. setting where low latency is crucial, latency is measured for all retrieval approaches. All latency measurements are made on a single machine using an Intel Core i7-1165G7 CPU. Queries used in this experiment are acquired by randomly sampling 100 queries from the FiQA-2018 dataset and are kept the same across approaches. The retrieval depth is set to 100 documents. Additionally, all latency measurements are conducted using the timeit[2] Python module with 100 loops and 7 runs for each step of the pipeline. Each step is measured separately to be able to identify the bottlenecks as well to ensure comparability.

Interpolation-based re-ranking pipeline consists of first-stage retrieval of 100 documents from the whole corpus by BM25 [7] followed by re-ranking of selected 100 candidates by TCT-COlBERT [23] and interpolation of scores to obtain the final score. The whole pipeline is implemented in PyTerrier with the FAST-FORWARD indexes package.

Dense retrieval pipeline consists of retrieving 100 documents from the whole corpus by TCT-ColBERT implemented in Pyserini using a flat FAISS index.

Hybrid Retrieval pipeline is reported using the BM25 for retrieval for 100 documents and TCT-ColBERT for retrieval of 100 documents from the whole collection. Consequently, the interpolation step is implemented as a manipulation of TREC run files generated by BM25 and TCT-COlBERT from the previous step. Furthermore, in hybrid retrieval all techniques for dealing with missing scores are measured - dropping document score as well as imputing zero, median and average across both original and normalized scores.

[2] https://github.com/python/cpython/blob/3.12/Lib/timeit.py



(a) FiQA-2018

(b) NF-Corpus

(c) TREC DL 2019

Figure 1: Tuning hyperparameter alpha for hybrid retrieval across datasets. Alpha values are reported with their corresponding nDCG@10 for both original (blue) and normalized scores (red)

4 Results

In this section we compare interpolation-based re-ranking with dense and hybrid approaches. Each section corresponds to one of the research question outlined in the Section 1 of this paper.

4.1 What is the importance of the lexical component in hybrid retrieval models and interpolation-based re-ranking?

In the combination of BM25 [7] and TCT-ColBERT [23] for hybrid and interpolation-based retrieval, the alpha value symbolizes the importance of the lexical component. This stems from the interpolation equation, as with a higher α value the lexical score provided by the sparse retriever will have bigger impact on the final score used for ranking.

In Table 2 we report the value of α hyperparameter across different datasets. It was obtained by conducting a grid search from the α inrange [0.05, 0.1, 0.25, 0.5, 0.75, 0.9] on the respective development sets. For FiQA-2018 and NF Corpus when the scores are normalized to offset the possible differences in magnitude, both sparse and dense scores have equal importance leading to an α of 0.5 for hybrid retrieval. For MS MARCO, there is higher importance placed on the dense score by reducing the α value in half compared to other datasets. The prevalence of the semantic score on TREC-DL-Psg'19, but not on other datasets could be explained by the training process of TCT-ColBERT, where MS MARCO is used for fine-tuning. The lexical component is relevant for the normalized scores, nevertheless the semantic score dominates. In addition, the identical α values for original scores in hybrid retrieval and interpolation-based re-ranking show, that the importance of lexical component does not depend on the retrieval approach, but rather on the models utilized.

Additionally, we report the results of hybrid retrieval hyperparamater tuning to showcase the effect of normalizing the score. As seen in Figure 1, the normalized scores are not able to improve the nDCG significantly compared to original scores, showing little benefit. Instead we recommend adjusting the range of α in hyperparameter tuning based on the differences

	FiQA-2018	**NF Corpus**	**TREC-DL-Psg'19**
Interpolation	0.1	0.1	0.05
Hybrid Retrieval			
↪original scores	0.1	0.1	0.05
↪normalized scores	0.5	0.5	0.25

Table 2: Alpha parameter values

in scale. Normalizing the score might allow for more even increments for the range of alpha values, however it also increases latency during runtime.

4.2 To what extent do missing document scores impact ranking performance in hybrid retrieval models?

To determine the missing document score impact in hybrid retrieval, we consider ranking performance and latency.

	FiQA-2018			**NF Corpus**			**TREC-DL-Psg'19**		
	nDCG$_{10}$	R$_{100}$	RR$_{10}$	nDCG$_{10}$	R$_{100}$	RR$_{10}$	nDCG$_{10}$	R$_{100}$	RR$_{10}$
Hybrid Retrieval									
↪ original scores									
↪ drop	0.313	0.625	0.379	0.329	0.243	0.535	0.691	0.566	0.808
↪ zero	0.313	**0.627**	0.379	**0.330**	0.273	0.533	**0.705**	**0.615**	0.831
↪ median	0.307	0.594	0.373	0.327	0.278	0.532	0.697	0.586	0.804
↪ average	0.306	0.590	0.372	0.326	0.279	0.529	0.693	0.577	0.797
↪ normalized scores									
↪ drop	**0.314**	0.624	**0.381**	0.329	0.243	0.535	0.688	0.561	0.821
↪ zero	0.280	0.608	0.343	0.326	0.267	**0.536**	0.655	0.585	**0.861**
↪ median	0.309	0.593	0.375	0.328	**0.280**	0.535	0.692	0.574	0.818
↪ average	0.308	0.585	0.374	0.327	0.280	0.532	0.687	0.566	0.818

Table 3: Ranking Performance for different missing score techniques for hybrid retrieval. Retrievers BM25 and TCT-ColBERT use depths $k_S = 1000$ and $k_D = 1000$.

Table 3 shows the ranking performance for all missing document techniques implemented for hybrid retrieval. Overall, it seems that best ranking performance across all datasets is achieved by using original scores for interpolation with substituting zero for missing document scores. Additionally, retaining the document that is not retrieved by other model yields significant improvements for the ranking performance in terms of recall. This is most

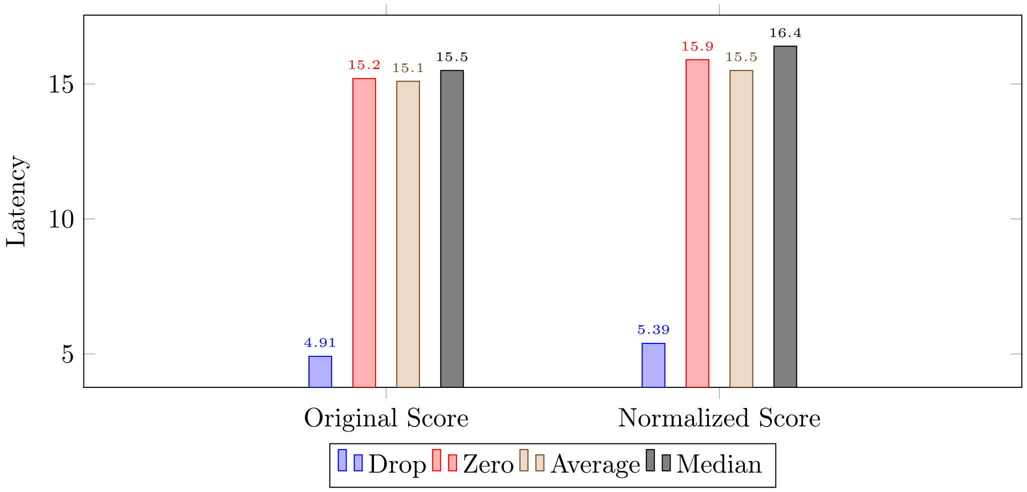


Figure 2: Interpolation latency results for FiQA-2018 for hybrid retrieval approaches. Latency is reported in miliseconds for both normalized and original scores.

likely due to models mitigating their weaknesses, thereby increasing recall rather than the top 10 documents ranking. Imputing the average or median does not seem to improve the ranking performance, especially when considering RR and nDCG. This is likely due to the fact that the average or the median score values can be quite high and if only one of the approaches retrieved the document for depth 1000 for a single query, it is unlikely that it could improve the top 10 document ranking.

Looking at the latency reports for interpolation of missing scores in Figure 2, the results are as expected. All results for normalized scores are slightly higher due to the normalization overhead. Dropping the document with missing score is much faster than any imputation techniques, as there is no need to scan the whole dataframe containing the rankings from the previous retrieval step. Differences in latency between imputation techniques are negligible, therefore the suitability can be determined mostly by ranking performance.

Based on the presented results, the best hybrid approach is keeping the original score with imputing zero in case of a missing document score. Normalization of the score offers little to no benefit and different score scale for both models can be mostly mitigated with hyperparameter tuning as demonstrated in Section 4.1. While dropping the document score can lead to benefits in certain domains as demonstrated on FiQA-2018, imputing zero is better suited for ad hoc retrieval, even with the additional latency.

4.3 How does interpolation-based re-ranking compare to dense and hybrid retrieval approaches in terms of ranking performance and latency?

In Table 4 we report the ranking performance of interpolation-based re-ranking with dense and hybrid ranking approaches with the original scores and imputing zero for missing document scores.

First, we observe that a purely dense approach is never able to achieve the best performance for any dataset included in the experiment. Consequently, we can conclude that

	FiQA-2018			**NF Corpus**			**TREC-DL-Psg'19**		
	nDCG$_{10}$	R$_{100}$	RR$_{10}$	nDCG$_{10}$	R$_{100}$	RR$_{10}$	nDCG$_{10}$	R$_{100}$	RR$_{10}$
Interpolation									
BM25 » TCT-ColBERT	**0.316**	**0.632**	**0.385**	**0.334**	0.254	**0.538**	0.693	0.585	0.808
Dense Retrieval									
TCT-ColBERT	0.265	0.561	0.322	0.267	0.250	0.464	0.670	0.565	0.820
Hybrid Retrieval									
BM25 + TCT-ColBERT	0.313	0.627	0.379	0.330	**0.273**	0.533	**0.705**	**0.615**	**0.831**

Table 4: Ranking Performance. Retrievers use depths $k_S = 1000$ (sparse) and $k_D = 1000$ (dense) with hybrid retrieval reportes with original scores and imputing zero for missing document scores.

incorporating a lexical component always improves ranking performance for the same dense model across different approaches. It complements the findings by Wang et al. [28] that the interpolation of BERT-based retrievers and sparse retrieval methods can boost the performance.

Furthermore, interpolation-based re-ranking outperforms the dense and hybrid approaches nDCG@10 and RR@10 in the smaller out-of-domain datasets. It goes to show that limiting the documents to set of candidates retrieved by BM25 [7], does not negatively affect the ranking of the relevant documents. However, NF Corpus recall (R) is negatively affected as can be seen in the 3% difference between interpolation-based re-ranking and the hybrid approach. This stems from recall being the only metric calculated at retrieval depth of 100 and discarding the rank of relevant documents in its calculation. Moreover, it suggests that while interpolation-based re-ranking performs better than dense retrieval, utilizing documents retrieved by TCT-ColBERT in first-stage retrieval is indeed beneficial and leads to more relevant documents in top 100 results.

Examining the ranking performance for MS MARCO, there is a steep increase in nDCG and RR compared to other datasets. The difference likely lies in the small number of relevant documents compared to the size of the whole corpus. It can not be accredited to the different domains, as both FiQA-2018 and NF Corpus have non-overlapping domains, but still achieve nDCG of approximately 0.3 compared to 0.7 for MS MARCO. Looking at MS MARCO, hybrid retrieval achieves best performance across all metrics, showing that employing a dense model for retrieval of documents from the whole corpus can bring ranking performance benefits, as the deeper understanding of context can lead to better relevance assessment of documents in a setting with small signal to noise ratio.

To evaluate the suitability of interpolation-based re-ranking in ad-hoc retrieval, it is crucial to consider latency, as ad-hoc retrieval has low-latency constraints. Figure 3 shows the latency measurements of all retrieval approaches, where each part of the retrieval pipeline was measured independently.

As expected, interpolation-based re-ranking has a clear advantage over the remaining approaches in end-to-end latency. The latency for utilizing TCT-ColBERT [23] is decreased

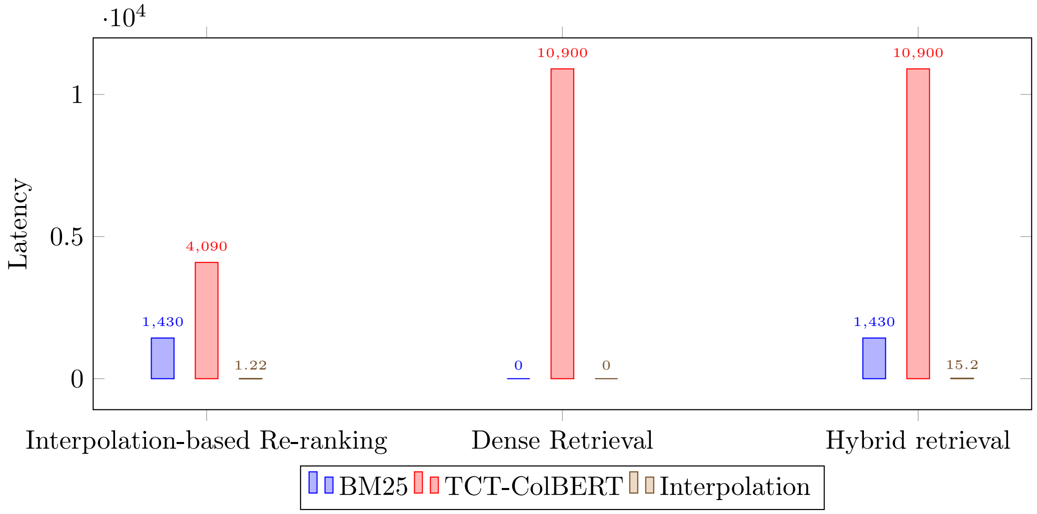


Figure 3: Latency results for 100 queries from FiQA-2018. Latency is reported in miliseconds for all stages - first-stage retrieval, re-ranking and interpolation across all retrieval approaches. Hybrid retrieval is reported for original scores with zero imputation

compared to the dense and hybrid approach. TCT-ColBERT is only used as a re-ranker in the second stage for interpolation-based re-ranking compared to retrieval of 1000 documents from the whole corpus as in both dense and hybrid setting. Hybrid retrieval has the highest end-to-end latency as it utilizes both a dense and a sparse model for first-stage retrieval.

Based on the presented results, interpolation-based re-ranking is most suitable for settings with low-latency constraints. Moreover, it achieves best ranking results for out-of-domain datasets such as FiQA-2018. On the other hand, hybrid retrieval outperforms interpolation-based re-ranking on MS MARCO, but comes with a cost of nearly double per-query latency. Therefore, it can be concluded that interpolation-based re-ranking is the best approach if low latency has higher priority than ranking performance, with hybrid retrieval prefferred for ad-hoc retrieval ranking performance. Dense retrieval is outperformed by both approaches and therefore is not recommended in this setting.

5 Limitations and Future Work

Due to the time constraints, it was not possible to conduct reliable statistical significance testing. As a consequence, the conclusions presented in this paper are mild in nature. Furthermore, the ranking experiment in this study can be extended to more dense models with different architecture choices such as cross-encoders. This would allow for a more complex comparison as the only dense model utilized in this paper is TCT-ColBERT. Aside from incorporating multiple dense models, the choice of sparse retrieval should be investigated in more detail. By incorporating state-of-the-art lexical approaches, there is potential for interpolation-re-ranking to outperform hybrid retrieval both in terms of latency and ranking performance. Last but not least, the missing scores for hybrid retrieval should be investigated for stand-alone hybrid models to clarify whether the benefits of imputing zero pertain across models, or whether they're specific to the combination of TCT-ColBERT with BM25.

6 Responsible Research

In the spirit of scientific integrity and responsible research, this paper aims to keep all the experiments fully compliant with the FAIR Principles [29]. This section outlines how the principles were taken into consideration throughout writing of the paper.

As this study is focused on the comparison of different retrieval approaches, it was essential to keep the experiment conditions identical to be able to draw any sound conclusions. This was achieved by the usage of BM25 [7] and TCT-ColBERT [23] across all approaches. Both interpolation-based re-ranking and hybrid retrieval used BM25 scores generated using the PyTerrier library[22]. This was further enforced by using the same encoder `'castorini/tct_colbert-msmarco'` for all three approaches.

Aside from the conditions of the retrieval approaches, the datasets used in the experiments were deliberately chosen for full transparency and accessibility. Both FiQA-2018 [25] and NF Corpus[26] are part of the BEIR[6] evaluation benchmark for information retrieval models, therefore making them suitable for evaluating performance of retrieval approaches. MS MARCO [5] completes the diversity of the chosen datasets by its large collection of passages. Its suitability is further supported by its use in evaluating the state-of-the-art approaches in 2019 at TREC conference. All of the datasets are publicly available making it possible to reproduce all of the experiments in their entirety. By reusing available bench marking datasets we ensure the reuse of digital assets.

Even though the libraries are open-source and datasets are publicly available, in an effort to make the verification possible to parties without access to specific resources such as a GPU needed for the indexing stage, or high-capacity RAM, there is a public repository on GitHub.com [3] containing all the run files in TREC format. This makes it possible to use Python packages such as ir_measures[4] to recalculate all the metrics reported in this paper and ensure their credibility. Alongside the generated run files, there are also examples of source code for running the experiments. By making it accessible to the general public, it can be inspected by the scientific community and help other researchers in this area to conduct similar experiments in the future. All the measures mentioned above make this paper FAIR compliant and allow for verifying integrity of reported results.

7 Conclusion

In this paper we compare the interpolation-based re-ranking to hybrid and dense approaches. We show that interpolation-based re-ranking is suitable for low-latency environments as a result of limiting the set of candidates in the re-ranking stage for semantic re-ranker. It outperforms dense retrieval in all domains, both in terms of latency and ranking performance. We further show that while hybrid retrieval has higher latency compared to interpolation-based re-ranking, it can outperform other approaches in terms of ranking performance due to the first-stage dense retriever. On the account of missing document scores, we show that normalizing the scores leads to little benefit, with imputing zero emerging on top for best ranking performance in hybrid retrieval. Last but not least, the results of dense retrieval, and parameter tuning for hybrid retrieval underline the importance of the lexical component in ad-hoc retrieval.

[3] `https://github.com/Buca11/RP-neural-rankers/tree/main`

[4] `https://ir-measur.es/en/latest/`

References

[1] Statista. Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025, 2024. Accessed: 2024-06-10.

[2] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W. Bruce Croft, and Xueqi Cheng. A deep look into neural ranking models for information retrieval. *Information Processing & Management*, 57(6):102067, November 2020.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *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, June 2019. Association for Computational Linguistics.

[4] Jurek Leonhardt, Henrik Müller, Koustav Rudra, Megha Khosla, Abhijit Anand, and Avishek Anand. Efficient neural ranking using forward indexes and lightweight encoders. *ACM Trans. Inf. Syst.*, 42(5), apr 2024.

[5] 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. Ms marco: A human generated machine reading comprehension dataset, 2018.

[6] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[7] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

[8] Yuanhua Lv and Chengxiang Zhai. Lower-bounding term frequency normalization. In *CIKM'11 - Proceedings of the 2011 ACM International Conference on Information and Knowledge Management*, International Conference on Information and Knowledge Management, Proceedings, pages 7–16, 2011. 20th ACM Conference on Information and Knowledge Management, CIKM'11 ; Conference date: 24-10-2011 Through 28-10-2011.

[9] Andrew Trotman, Xiangfei Jia, and Matt Crane. Towards an efficient and effective search engine. In *OSIR@SIGIR*, 2012.

[10] Zhuyun Dai and Jamie Callan. Context-aware sentence/passage term importance estimation for first stage retrieval, 2019.

[11] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. Splade: Sparse lexical and expansion model for first stage ranking, 2021.

[12] Jimmy Lin and Xueguang Ma. A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques, 2021.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[14] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.

[15] J. Wang et al. Utilizing bert for information retrieval: Survey, applications, resources, and challenges. *arXiv*, Feb 2024. Accessed: May 25, 2024. [Online]. Available: `http://arxiv.org/abs/2403.00784`.

[16] Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert, 2020.

[17] Yinfei Yang, Ning Jin, Kuo Lin, Mandy Guo, and Daniel Cer. Neural retrieval for question answering with cross-attention supervised data augmentation, 2020.

[18] Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. Complementing lexical retrieval with semantic residual embedding, 2021.

[19] Luyu Gao, Zhuyun Dai, and Jamie Callan. Coil: Revisit exact lexical match in information retrieval with contextualized inverted list, 2021.

[20] Euna Jung, Jaekeol Choi, and Wonjong Rhee. Semi-siamese bi-encoder neural ranking model using lightweight fine-tuning. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 502–511, New York, NY, USA, 2022. Association for Computing Machinery.

[21] Stephen Robertson. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation - J DOC*, 60:503–520, 10 2004.

[22] Craig Macdonald, Nicola Tonellotto, Sean MacAvaney, and Iadh Ounis. Pyterrier: Declarative experimentation in python from bm25 to dense retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 4526–4533, New York, NY, USA, 2021. Association for Computing Machinery.

[23] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. Distilling dense representations for ranking using tightly-coupled teachers, 2020.

[24] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362, 2021.

[25] Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. Www'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1941–1942, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.

[26] Vanya Boteva, Danial Gholipour, Aleksandr Sokolov, and Stefan Riezler. A full-text learning to rank dataset for medical information retrieval. In Nicola Ferro, Fabio Crestani, Marco de Gemmis, Maria Maistro, Viviana Patti, Fabrizio Silvestri, and Gianmaria Silvello, editors, *Advances in Information Retrieval. ECIR 2016*, volume 9626 of *Lecture Notes in Computer Science*, pages 625–630. Springer, Cham, 2016.

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

[28] Shuai Wang, Shengyao Zhuang, and Guido Zuccon. Bert-based dense retrievers require interpolation with bm25 for effective passage retrieval. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '21, page 317–324, New York, NY, USA, 2021. Association for Computing Machinery.

[29] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Julien Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jaime S. Grethe, Jildau A. Groenhof, Carole G. Groth, Antony S. Jacobsen, Morris A. Jeffery, Pankaj S. Joo, Gerhard Kamp, Menno de Kok, Ulrich Kuhn, Jack A. Kok, Robert Koureas, Joachim L. Kucera, Jay Jay K. Lee, Barend Mons, Berend R. Monach, Daniele Natale, Marek K. Pfisterer, Henning L. Plomp, Philippe Rocca-Serra, Marco Roos, Rene S. van Schaik, Susanna Sansone, Erik Schultes, Stephen S. Sengstag, Cesar N. Smith, Alexander H. Spek, Maja R. Suarez-Figueroa, Jan Velterop, Arjan de Waard, Mark D. Wilkinson, Albert W. Williams, Xin Zhao, and James A. Smith. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3:160018, 2016.