Beyond Questions: Leveraging ColBERT for Keyphrase Search

Jorge Gabín[a,b,*], Javier Parapar[b] and Craig Macdonald[c]

[a]*Linknovate Science, Santiago de Compostela, Spain*

[b]*IRLab, CITIC, Computer Science Department, University of A Coruña, A Coruña, Spain*

[c]*University of Glasgow, Glasgow, Scotland*

ABSTRACT

While question-like queries are gaining popularity and search engines' users increasingly adopt them, keyphrase search has traditionally been the cornerstone of web search. This query type is also prevalent in specialised search tasks such as academic or professional search, where experts rely on keyphrases to articulate their information needs. However, current dense retrieval models often fail with keyphrase-like queries, primarily because they are mostly trained on question-like ones. This paper introduces a novel model that employs the `ColBERT` architecture to enhance document ranking for keyphrase queries. For that, given the lack of large keyphrase-based retrieval datasets, we first explore how Large Language Models can convert question-like queries into keyphrase format. Then, using those keyphrases, we train a keyphrase-based `ColBERT` ranker (`ColBERTKP`$_{QD}$) to improve the performance when working with keyphrase queries. Furthermore, to reduce the training costs associated with training the full `ColBERT` model, we investigate the feasibility of training only a keyphrase query encoder while keeping the document encoder weights static (`ColBERTKP`$_Q$). We assess our proposals' ranking performance using both automatically generated and manually annotated keyphrases. Our results reveal the potential of the late interaction architecture when working under the keyphrase search scenario. This study's code and generated resources are available at https://github.com/JorgeGabin/ColBERTKP.

1. Introduction

The search process is increasingly adopting question-like queries (Dalton et al., 2022; GUO et al., 2022; Khurana et al., 2023). This shift is driven by the enhanced ability of search engines to better process such longer queries and the growing influence of conversational models, which encourage users to phrase their searches as questions. Despite this trend, keyphrase search remains essential, especially in specialised fields such as academic and professional search (Jacsó, 2015; Li et al., 2017; Russell-Rose et al., 2018; Lahiri et al., 2024). Experts often use keyphrases and boolean queries to express their information needs. However, nowadays most state-of-the-art neural retrieval models are trained primarily on datasets with question-like queries, such as those from MSMarco (Nguyen et al., 2016).

Here, we focus on the persistent importance of keyphrase queries and propose training dense retrieval models specifically for this query format. Transformer-based (Vaswani et al., 2017) models like `BERT` (Devlin et al., 2019) and, specifically, alternatives like `DPR` (Karpukhin et al., 2020), `ANCE` (Xiong et al., 2021), `ColBERT` (Khattab and Zaharia, 2020), `Col★` (Wang et al., 2023), `GTR` (Ni et al., 2022), or `XTR` (Lee et al., 2023), have demonstrated the effectiveness of dense retrieval models for ranking. While traditional methods use inverted indexes, dense retrieval represents documents and queries as dense vectors, which are then matched using approximate nearest neighbours (ANN) algorithms, such as those present in the `FAISS` toolkit (Douze et al., 2024; Johnson et al., 2021).

As mentioned above, current dense retrieval models are predominantly trained on question-like queries due to the availability of large-scale datasets covering this specific query type. We argue that this has inadvertently made queries formulated as questions the standard for these models, often overlooking other query formats. Indeed, our observations indicate that traditional dense models might underperform with keyphrase queries. For instance, in Table 1 we can observe the difference between question-like and keyphrase-like queries. While both models retrieve a highly relevant document for the question-like query, `ColBERT` struggles with the keyphrase query while one of our keyphrase-tailored models (`ColBERTKP`$_Q$) succeeds.

---

*Corresponding author

✉ jorge.gabin@udc.es (J. Gabín); javier.parapar@udc.es (J. Parapar); craig.macdonald@glasgow.ac.uk (C. Macdonald)

ORCID(s): 0000-0002-5494-0765 (J. Gabín); 0000-0002-5997-8252 (J. Parapar); 0000-0003-3143-279X (C. Macdonald)

Table 1

Highest ranked passages by ColBERT and ColBERTKP$_Q$ models for a TREC 2019 DL query in both the original and keyphrase format. The "Label" column contains the assessment of that document for that query in the qrel file, with − denoting unjudged.

Doc.	System	Top-1 Passage	Label
962179: when was the salvation army founded			
5653659	ColBERT	The Salvation Army was founded in London's East End in 1865 by one-time Methodist Reform Church minister William Booth and his wife Catherine. Originally, Booth named the organisation the East London Christian Mission.	3
2978866	ColBERTKP$_Q$	The Salvation Army was founded in London's East End in 1865 by one-time Methodist Reform Church minister William Booth and his wife Catherine. Originally, Booth named the organisation the East London Christian Mission.	3
962179: salvation army foundation year			
5642478	ColBERT	February 24, 2011. In addition to money, the Salvation Army accepts donations of vehicles, furniture, appliances, clothing and countless other household items. As a religious and charitable organization, the Salvation Army resells your donations to fund its adult rehabilitation programs. Because charitable donations are tax-deductible, the organization provides a receipt as documentation.	-
2329699	ColBERTKP$_Q$	History. The Salvation Army began in 1865 when William Booth, a London minister, gave up the comfort of his pulpit and decided to take his message into the streets where it would reach the poor, the homeless, the hungry and the destitute.	3

In this paper, we hypothesise that training those dense retrieval models with a keyphrase-focused dataset may alleviate the problems of current state-of-the-art models when facing this type of query. Keyphrase-tailored models may partially solve the applicability of dense re-ranking in professional, academic, and even web searches, where short keyphrase-like queries continue to be the prevailing query type (Silverstein et al., 1999; Reimer et al., 2023; Taghavi et al., 2012). Moreover, recognising the computational costs associated with training the full ColBERT model, we investigate the feasibility of training a keyphrase query encoder while keeping the document encoder weights frozen.

Training neural ranking models for keyphrase search requires sufficient training data. However, previous studies (Bolotova et al., 2022) indicate that the commonly used MSMarco dataset for training neural ranking models primarily consists of question-like queries. In fact, following the Bolotova et al. (2022) strategy, only 0.40% of the queries are classified as non-questions. Therefore, we propose using generative models to reformulate question-like queries into keyphrase format to address the lack of keyphrase-style datasets.

The contributions of this paper are: (1) We demonstrate the effectiveness of keyphrase-based models in keyphrase search scenarios; (2) we discuss the effect of only training the query encoder versus training both a query and a document encoder; (3) we propose a method to generate keyphrase queries from questions or other query types using Large Language Models (LLMs); (4) we show whether the proposed keyphrase-based training generalises to other models; (5) we manually curate keyphrase queries for the TREC 2019 DL Track test queries to verify the validity of the automatically generated keyphrases; (6) we study the main differences between existing dense retrieval models and our proposals; (7) we assess the performance of both the original model and its keyphrase-enhanced versions within a more realistic, mixed-query scenario; (8) finally, we conduct experiments to evaluate the models' generalisability to different query types, focusing specifically on traditional title-based queries.

Our main finding is that ranking models trained in a dataset primarily composed of keyphrase queries outperform classical dense retrieval approaches in the keyphrase search task while showing comparable performance to original models on question-like, queries. Additionally, we show that the proposed training approach reduces the reliance on lexical matching by enabling the encoding of more relevant information within the special tokens. Finally,

our keyphrase-tailored models show improved generalisability compared to the original model, delivering better performance on traditional title-format query collections.

The remainder of this paper is structured as follows: Section 2 provides an overview of related work concerning dense retrieval and query and keyphrase generation methods. Our proposed training approach and automatic and manual resource generation process are elaborated upon in Section 3 and Section 4 respectively. The setup and results of our experiments are discussed in Section 5. Finally, we summarise our findings and future work in Section 6.

2. Related Work

Here, we review the state of the art in Pre-trained Language Model (PLM)-based retrieval (§ 2.1) and existing works in query and keyphrase generation (§ 2.2).

2.1. PLM-based Retrieval

Cross-encoder models specialise in evaluating sentence or text pair classification. They provide a score for input ⟨*query*, *text*⟩ pairs, indicating their relationship or similarity. These models have been successfully used in several tasks, with ranking being one where they have demonstrated exceptional effectiveness. However, because they require both the query and the document to be processed simultaneously, they are typically employed only in re-ranking scenarios due to their limited efficiency.

Cross-encoders were first introduced by Devlin et al. (2019) and applied in many NLP tasks over the years. Regarding retrieval tasks, one of the most well-known examples is `monoT5` (Nogueira et al., 2020), a re-ranking model that leverages the text generation capabilities of T5 (Text-to-Text Transfer Transformer) (Raffel et al., 2020) to determine whether a document is relevant to a given query. Later, Pradeep et al. (2021) presented `duoT5` as part of their re-ranking pipeline, a more robust yet resource-intensive model that evaluates the relevance of two documents for a specific query and selects the most relevant one. Indeed, because of its high computational requirements, `duoT5` is usually only employed to re-rank the very top (e.g. top-2) documents.

An alternative to overcome the high computational cost of cross-encoders is dense retrieval. Dense retrieval models leverage contextualised dense representations of both queries and documents to determine relevance scores, allowing both end-to-end and re-ranking strategies. Macdonald et al. (2021) compared the two existing representation strategies used in dense retrieval setups: single representation and multiple representation models.

In single representation approaches, exemplified by `DPR` (Karpukhin et al., 2020), `ANCE` (Xiong et al., 2021), and `TCT-ColBERT` (Lin et al., 2021), each query or document is encoded into a single embedding. Subsequently, the relevance between the query and document is determined by computing the dot-product of the encoded dense query and document vectors.

However, this study focuses on multiple representation models (Ni et al., 2022; Lee et al., 2023; Wang et al., 2023), exemplified by `ColBERT` (Khattab and Zaharia, 2020). The motivation behind this choice mirrors the industry's growing interest in `ColBERT` (Kim, 2022; Clavié, 2024; Lee et al., 2024; Jha et al., 2024; Intel, 2024). Models like `ColBERT` provide interpretability through late attention mechanisms, identifying the most relevant sections of a document in response to a given query. This is crucial in real-world scenarios where explainability is critical, as it enables users to comprehend the underlying decision-making process. Moreover, it proves invaluable in our research, facilitating the understanding of the subtle differences between question-like and keyphrase-like queries.

Recent work has focused on various fronts, such as distilling knowledge from high-performing models like `ColBERT` to improve single-representation dense retrieval models (Hofstätter et al., 2020, 2021; Lin et al., 2020; Wang et al., 2022); the impact of negative samples on training (Lin et al., 2021; Zhan et al., 2021), as selecting proper hard negative examples instead of random negatives improves dense retrieval models performance; embedding pruning and compression to reduce indices size (Lassance et al., 2022; Santhanam et al., 2022). However, to the best of our knowledge, we are the first to study the training and evaluation of dense retrieval models for keyphrase search.

2.2. Query and Keyphrase Generation

In this paper, we want to explore the adaptability of dense retrieval models to diverse query types, particularly keyphrase-like queries. For that, we need to create training and testing datasets for keyphrase search. Regarding that objective, we have to remark on the extensive use of generative models in the area.

A well-known example is the emergence of automatic keyphrase generation models, which are replacing traditional extraction-based approaches progressively. Generative models like `BART` (Kulkarni et al., 2022), T5 (Gabín et al., 2022)

and GPT (Song et al., 2023; Martínez-Cruz et al., 2023) have proved to outperform existing automatic keyphrase extraction methods, such as EmbedRank (Bennani-Smires et al., 2018) or KEA (Witten et al., 1999), in labelling documents with keyphrases. The primary advantage of these models over extractive approaches is their capability to generate keyphrases not explicitly present in the input text.

Another task where generative models have been used recently is query expansion. Until the appearance of these generation-based models, pseudo-relevance feedback (PRF) approaches (Wang et al., 2023a,b) dominated the query expansion task; these leverage documents retrieved by the initial queries to extract terms which are used to expand the queries. Recent studies (Jagerman et al., 2023; Wang et al., 2023) leveraged LLMs to generate documents or answers to queries before the ranking phase, aiming to expand the query with them to enhance final rankings.

Query generation techniques have also been used to generate query variants for test collections. Alaofi et al. (2023) proposed using LLMs to automatically generate query variants from a description of an information need. They employed GPT-3.5 to generate variants capable of addressing the information need outlined in the input backstory. Their study reveals a substantial overlap between documents retrieved using human-generated and AI-generated queries.

Generative models have also been used to generate queries to expand documents. For example, docT5query (Nogueira et al., 2019) involves the generation of queries that the input document could potentially answer. These predicted queries are appended to the original documents before indexing.

Nevertheless, as far as we are aware, no previous research has focused on transforming question-like queries into keyphrase-format queries. We believe this area deserves exploration to facilitate the generation of resources for training models tailored for keyphrase search scenarios, such as academic or professional search (Jacsó, 2015; Russell-Rose et al., 2018; Lahiri et al., 2024).

3. Training Keyphrase-based Rankers

Here, we first explain the functioning of ColBERT (§ 3.1), followed by its adaptation to keyphrase search scenarios (§ 3.2).

3.1. Preliminaries

In this section, we outline the training process of the ColBERT model, which serves as the backbone model for our approaches.

First, we can define ColBERT as a linear layer upon the raw token embeddings generated by a Transformer encoder (usually BERT), $ColBERT = Linear((BERT(t_1,..t_n), m)) \in \mathbb{R}^m$, where m is typically set to 128 (Khattab and Zaharia, 2020). So, the process of producing query (ϕ_q) and document (ϕ_d) embeddings using ColBERT can be expressed as:

$$\phi_q = ColBERT([CLS], [Q], q_1, ..., q_{|q|}) \in \mathbb{R}^{\mu x m}$$
$$\phi_d = ColBERT([CLS], [D], d_1, ..., d_{|d|}) \in \mathbb{R}^{|d| x m}$$

where μ represents the padded length of the tokenised query.

Then, given a query q and a document d, the scoring function for the ColBERT model is defined as:

$$score(q,d) = \sum_{i=1}^{|q|} MaxSim(\phi_{q_i}, \phi_d) = \sum_{i=1}^{|q|} \max_{j=1,...,|d|} Sim(\phi_{q_i}, \phi_{d_j})$$

where Sim represents the function used to compute the similarity between query and document embeddings.

The ColBERT model undergoes training using the MSMarco training triples, each triple comprising a query (q), a relevant passage (d^+), and a non-relevant passage (d^-). During training, the objective is to maximise the similarity between the query and relevant passages while minimising similarity with non-relevant passages.

Therefore, considering a triple $\langle q, d^+, d^- \rangle$, ColBERT computes a score for each document d^+, d^- w.r.t. q. This process is optimised using pairwise softmax cross-entropy loss over the computed scores of d^+ and d^-.

3.2. From Questions to Keyphrases

Given the example shown in Table 1, we argue that training dense retrieval models on datasets like MSMarco tends to bias them towards performing well on question-like queries. To address this limitation of current retrieval models, we propose two adaptations to the training process to better suit keyphrase search scenarios.
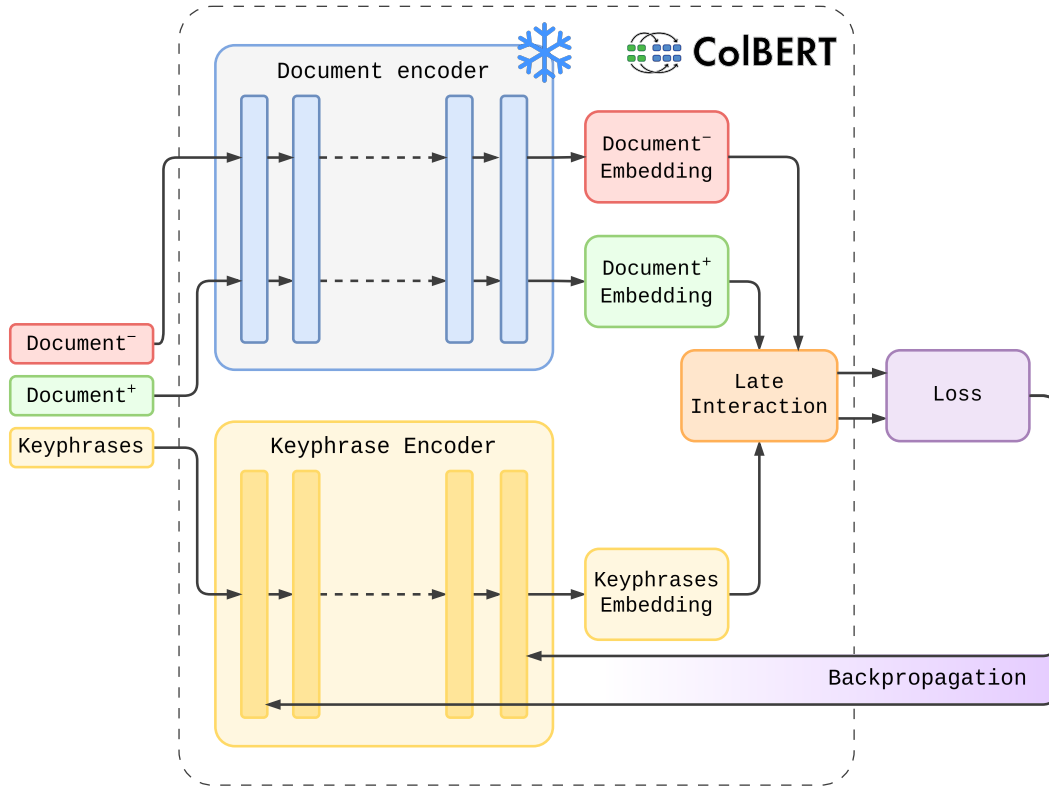
Figure 1: Training process of the `ColBERTKP`$_Q$ model using the transformed MSMarco training triples.

There are differences in matching behaviour between question-like queries and keyphrase ones that justify the need for training specific models. Question-like queries usually contain interrogative adverb phrases (e.g., *"when", "where", "how"*, etc.), which guide the model towards the information need and might match with answer words (e.g. *"why"* with *"because"*). However, those adverbs are absent in keyphrases, so models must learn to encode that information differently. Keyphrase-tailored models encode this kind of information by using the special tokens (e.g. `[CLS]`, `[Q]`, `[SEP]` and `[MASK]`) to solve the problem that existing models have.

In the first approach, we train both a keyphrase-like query encoder and a document encoder concurrently, following the standard `ColBERT` training strategy. This training process optimises the similarity between input keyphrases and relevant documents, while simultaneously minimising similarity with non-relevant ones. By jointly training these encoders, we aim to capture the relationships between keyphrases and documents, thereby improving the effectiveness of the document ranking process in keyphrase search scenarios. We name the resulting model `ColBERTKP`$_{QD}$.

The second alternative aims to mitigate computational costs associated with training the entire `ColBERT` model while leveraging its powerful capabilities. In this approach, we freeze the document encoder layer (using the trained weights of a `ColBERT` checkpoint) while focusing solely on training a keyphrase encoder. We refer to this encoder model as `ColBERTKP`$_Q$.

Figure 1 shows the training process of the `ColBERTKP`$_Q$ model, where transformed MSMarco triples (*keyphrase query, relevant document, non-relevant document*) are used as inputs. The model generates dense representations for the query and documents, then calculates scores to learn optimal ranking by maximising the score for the relevant document and minimising it for the non-relevant one. Finally, the weights of the keyphrase encoder are updated via backpropagation while the document encoder weights remain constant.

However, training and evaluating a keyphrase-based ranker model necessitates datasets comprising keyphrase queries. For that, we transform the queries in the MSMarco training set into keyphrase format. In Section 4.2, we detail how we leverage an LLM to construct datasets specifically tailored for keyphrase search. Then, instead of the conventional triplet structure, which includes a question, relevant passage, and non-relevant passage, our dataset now features a keyphrase query, along with the corresponding passages.

We formalised our training approaches as follows. First, analogously to ColBERT, the process of encoding a keyphrase query k and a document d using ColBERTKP$_{QD}$ can be formalised as in Eq. 1. On the other hand, we can define keyphrase query and document embeddings for the ColBERTKP$_Q$ model as in Eq. 2.

$$\phi_k = \texttt{ColBERTKP}_{QD}([\texttt{CLS}], [\texttt{Q}], k_1, ..., k_{|k|}) \in \mathbb{R}^{\mu x m}$$
$$\phi_d = \texttt{ColBERTKP}_{QD}([\texttt{CLS}], [\texttt{D}], d_1, ..., d_{|d|}) \in \mathbb{R}^{|d|x m} \tag{1}$$

Different to ColBERTKP$_{QD}$, which learns both the query and document encoders, ColBERTKP$_Q$ uses the original ColBERT model to encode the documents.

$$\phi_k = \texttt{ColBERTKP}_{Q}([\texttt{CLS}], [\texttt{Q}], k_1, ..., k_{|k|}) \in \mathbb{R}^{\mu x m}$$
$$\phi_d = \texttt{ColBERT}_{D}([\texttt{CLS}], [\texttt{D}], d_1, ..., d_{|d|}) \in \mathbb{R}^{|d|x m} \tag{2}$$

Then, the score function for ColBERTKP$_{QD}$ and ColBERTKP$_Q$ given a keyphrase query ($k = \texttt{Q2K}(q)$) and a document (d) is:

$$score_{kp}(k, d) = \sum_{i=1}^{|k|} MaxSim(\phi_{k_i}, \phi_d) = \sum_{i=1}^{|k|} \max_{j=1,...,|d|} Sim(\phi_{k_i}, \phi_{d_j})$$

Finally, we define a sequence-to-sequence model as:

$$\texttt{Q2K}(seq_{1..N}) \rightarrow seq_{1..M}$$

Specifically, Q2K transforms any query into keyphrase-format, e.g. Q2K(*"how to train a bi-encoder"*) = *"bi-encoder training"*.

Therefore, for a triple $\langle q, d^+, d^- \rangle$ akin to those employed to train ColBERT, we initially convert the query q into a keyphrase-like format using the model Q2K, denoted as $k = \texttt{Q2K}(q)$. Subsequently, for the triple $\langle k, d^+, d^- \rangle$, both ColBERTKP$_{QD}$ and ColBERTKP$_Q$ compute scores individually for each document. Finally, as previously mentioned, the scoring process is optimised using pairwise cross-entropy loss over the scores of d^+ and d^-.

While the loss computation remains consistent for both the ColBERTKP$_{QD}$ and ColBERTKP$_Q$ models, a disparity appears in their training procedures. In training the former, we optimise weights for both the document and query encoders. In contrast, for the ColBERTKP$_Q$ model, we exclusively apply backpropagation to the query encoder layers, leaving the document encoder unchanged throughout the training process.

Summing up, we present two training methodologies for dense retrieval models optimised for keyphrase search. These models excel in capturing the subtle relationships between keyphrase-style queries and documents, thereby mitigating the dependence on lexical matching in favour of prioritising special token matching. Moreover, the ColBERTKP$_Q$ model offers the advantage of compatibility with existing ColBERT indices. This flexibility allows for the utilisation of diverse query encoders based on query type without necessitating multiple document indices.

4. Resource Alignment for Keyphrase-based Retrieval

In this section, we explore the motivation (§ 4.1) behind developing datasets centred on keyphrases and outline the steps taken to construct them. We first show an automated approach (§ 4.2) and finish with the creation of a manually curated dataset (§ 4.3). Appendix A provides examples of both automatically generated and manually annotated keyphrases for selected TREC topics.

4.1. Motivation

Long-question-format queries serve as the standard for training and evaluating dense retrieval models. To showcase this fact, in Table 2 we show the results of classifying all the MSMarco (Nguyen et al., 2016) dev queries as whether they are questions. Resembling the work by Cambazoglu et al. (2021) where they present a multi-faceted taxonomy to classify questions asked in web search engines, we use a naive classifier that checks if the query starts with any of the following words: *"what"*, *"when"*, *"where"*, *"which"*, *"who"*, *"whom"*, *"whose"*, *"why"*, *"how"*, *"is"*, *"are"*, *"do"*, *"does"* as a baseline. We also report results using two BERT-based classification models specifically fine-tuned for the task and

Table 2

Percentage of MSMarco 101k dev queries labelled as questions and accuracy of each method w.r.t. the naive classification approach.

Method	Questions (%)	Accuracy w.r.t. Naive (%)
Naive	70.06	-
BERT v1[1]	76.05	93.37
BERT v2[2]	85.77	81.36
Mistral[3] (Jiang et al., 2023)	81.63	75.93

an LLM instructed to identify whether a query is a question. The accuracy of these methods is assessed against the naive classifier, which may exhibit a few false negatives but is generally conservative in producing false positives.

The results in Table 2 reveal that even the less restrictive approach, represented by the naive classifier, categorises over 70% of the MSMarco queries as questions. Furthermore, both the BERT-based and LLM-based methods exhibit strong agreement with the classifications made by the naive classifier.

4.2. Automatically Generating Keyphrase Queries

Before tackling the training and evaluation of keyphrase-based dense ranking models, it is necessary to have training and testing datasets tailored specifically for keyphrases. However, as previously mentioned, these datasets are scarce. Given this fact, we devise a strategy to transform any query into a keyphrase-format query.

We explore two different approaches for this task. The first strategy employs a T5 model fine-tuned for the automatic keyphrase generation (AKG) task. While keyphrase generation models traditionally process documents to label them with keyphrases, we hypothesise that they may be able to generate keyphrases for queries effectively.

The second approach involves leveraging an LLM specifically instructed to generate keyphrase-format queries given any query. We design the guidelines to direct the LLM's query generation process and complement the prompt with some in-context examples. This prompt can be found in Appendix B and within the code in the repository.

Preliminary experiments reveal that using T5-generated keyphrases as queries leads to a significant decline in the models' ranking performance compared to the original queries. Conversely, testing with LLM-generated keyphrases demonstrates comparable performance against the original queries and even some instances of improvement, particularly when employing keyphrase-based ranking models. The primary difference between these two models lies in the number of generated keyphrases. The T5-based model typically generates a larger set of keyphrases for each query, which aligns with its training process. On the other hand, the LLM-based approach tends to be more concise, often producing either a single keyphrase or a small set thereof. Given the superior performance of the LLM-based method, we employ it to generate training and test data automatically.

However, we believe that solely relying on LLM-generated data to evaluate the performance of the models is insufficient. First, there is a risk of bias in the evaluation process, as the similarity between the training and testing queries—both generated by the same model—could affect the results. Second, LLM-generated queries may not accurately reflect real-world keyphrase queries.

4.3. Manually Curating a Test Set

We introduce a manually curated keyphrase-based query set for the TREC DL 2019 dataset to assess the validity of the automatically generated keyphrases. This test set is crafted by three computer science PhDs with experience in the information retrieval field. Their expertise ensures a thorough formulation of queries.

We abstain from providing specific guidelines on query formulation to keep each user's autonomy in query formulation. Users are tasked with generating keyphrase-format queries to address the information needs outlined in the TREC queries. The only directive provided is that they can employ one or multiple keyphrases. We instruct users to separate keyphrases (when using more than one) with commas to adhere to the LLM format.

---

[1]shahrukhx01/question-vs-statement-classifier
[2]bert-mini-finetune-question-detection
[3]mistralai/Mistral-7B-Instruct-v0.2

To facilitate the annotation process, we opt for a straightforward approach. We provide each assessor with a two-column sheet, where one column contains the TREC queries and the other is blank for writing the keyphrase queries. Counting on a manually labelled dataset helps us verify the validity of the automatically generated keyphrase queries and also ensures the reliability of the experiments conducted using them.

5. Experiments

We structure our experiments to address the following research questions:

RQ1 Which keyphrase-based training approach (full or encoder-only) performs better in keyphrase search scenarios?

RQ2 Is the performance of keyphrase-based models degraded when using original queries?

RQ3 Does the keyphrase-based training generalise to other models?

RQ4 Does the effectiveness observed with automatically generated keyphrases extend to manually annotated ones?

RQ5 How are keyphrase models and queries different from original queries and models?

RQ6 How effectively do the proposed models adapt to mixed query scenarios?

RQ7 Does the effectiveness of keyphrase-tailored models translate to traditional keyword (title-only) queries?

5.1. Experimental Setup
5.1.1. Datasets

We perform experiments on the MSMarco passage corpus, utilising queries from the TREC 2019 and TREC 2020 DL tracks, as well as queries from the MSMarco dev set. Nevertheless, our primary objective is to evaluate the effectiveness of models in keyphrase search scenarios. Therefore, alongside assessing the models' performance on the datasets' original queries, we explore their capability with automatically generated keyphrase-like queries, derived by transforming the original queries into keyphrases. Furthermore, we conduct experiments using manually labelled keyphrases for the TREC 2019 DL queries.

Regarding the training data for the keyphrase-based models, we transform the MSMarco train triples using the LLM-based approach introduced in Section 4.2, translating the queries into keyphrase format. Therefore, the dataset we use to train our models is solely formed by keyphrase queries and the corresponding passages.

We also conduct experiments using traditional title-based query collections, employing three query sets—TREC Robust 2004 (Voorhees, 2005), TREC 7 (Hawking et al., 1998), and TREC 8 (Hawking et al., 1999)—all of which are based on documents from TREC disks 4 and 5.

5.1.2. Metrics

We employ commonly used metrics for the TREC 2019 DL and TREC 2020 DL tracks query sets, as outlined in the corresponding track overview papers (Craswell et al., 2020a,b). Specifically, following common practices, we report the Mean Reciprocal Rank (`MRR`) and normalised Discounted Cumulative Gain (`nDCG`) calculated at rank cutoff 10, as well as Mean Average Precision (`MAP`) at rank cutoff 1000. For the MSMarco dev query set, we report `MMR@10` and Recall (`R`) at rank cutoffs 50, 200 and 1000.

For experiments on traditional title-query collections, we report both `MAP` at rank cutoff 1000 and `nDCG` at rank cutoff 10.

5.1.3. Baselines

In our experiments, we compare the effectiveness of our keyphrase-based models with `ColBERT`, which serves as the backbone of our work. We use the checkpoint provided by Wang et al. (2023) in their reproducibility study, specifically, we employ the cosine similarity version. Additionally, we include results obtained from `BM25` as a reference. Finally, we examine the performance of one widely recognised re-ranking strategy when experimenting with the generalisation of the proposed training approach: `monoT5`[4].

---

[4]castorini/monot5-base-msmarco

5.1.4. Implementation & Settings

We conduct all the experiments using PyTerrier (Macdonald and Tonellotto, 2020) on an Nvidia A100 GPU.

Our training procedure for `ColBERT`-based models follows the methodology outlined by Wang et al. (2023) in their reproducibility study. Beginning with the baseline checkpoint, we train the model with the same number of examples as in (Wang et al., 2023), up to 25k steps with a batch size of 128. Consistently, we employ cosine similarity as our similarity function. Additionally, we experimented with training the keyphrase-based models using `bert-base` as the initial checkpoint but observed inferior results (not included in the paper for brevity). Regarding parameters configuration, following the original `ColBERT` paper (Khattab and Zaharia, 2020), we set the query padded length (μ) to 32 and the embedding size (m) to 128.

For the re-ranking pipelines (indicated by: BM25 » $\mathcal{M}$), we re-rank the top 1000 documents retrieved by BM25 using the model $\mathcal{M}$.

Regarding statistical significance tests, we compare ranking strategies that belong to the same category and only the keyphrase-format queries: re-rankers with keyphrase queries and end-to-end models with keyphrase queries. By narrowing down our comparisons, we aim to provide more precise insights into the effectiveness of different ranking approaches and models across each search scenario. We use the paired t-test ($p \leq 0.05$) and apply the Holm-Bonferroni multiple testing correction.

For original queries, where our goal is that our proposals achieve equivalent results to existing models rather than surpassing them, we employ the Two One-Sided Test (TOST) (Schuirmann, 1987). In this case, we also compare ranking strategies within the same category, using lower and upper bounds of -0.01 and 0.01, respectively, with a significance level set at 0.05.

For the experiments on title-based queries, the documents from TREC disks 4 and 5 contain full text rather than passages. Due to the limitations of BERT-based models, which restrict input to 512 tokens, we segment the full text into passages with a length of 128 and a stride of 64. Then, the score of a document is computed by taking the score of its highest ranked passage, a.k.a., its *max passage*.

Finally, concerning the automatic keyphrase generation process, we employ two distinct approaches for generating the training and test datasets. As detailed in Section 4.2, we use an LLM for this task, we specifically leverage an instruction fine-tuned version of `Mistral`[5] (Jiang et al., 2023). For the generation of the training dataset, given the substantial volume of queries to process, we employ an efficient 4-bit precision inference approach (Dettmers and Zettlemoyer, 2023), while original precision is used for building the keyphrase test query sets.

5.2. Results and Analysis

In this section, we report and discuss results to answer all the research questions.

5.2.1. RQ1. Which keyphrase-based training approach (full or encoder-only) performs better in keyphrase search scenarios?

To address our initial research question, we present the ranking performance of both proposed models (`ColBERTKP`$_{QD}$ and `ColBERTKP`$_Q$), using the automatically generated keyphrases for TREC DL and MSMarco dev queries (as presented in Section 4.2) and reporting outcomes for both end-to-end ranking and re-ranking strategies.

Table 3 reports the performance of the proposed models on automatically generated keyphrase queries. These results show the dominance of keyphrase-based models in the keyphrase search scenario. Both `ColBERTKP`$_{QD}$ and `ColBERTKP`$_Q$ outperform the baselines across end-to-end and re-ranking evaluation setups, demonstrating significant differences, particularly evident in the MSMarco Dev Small query set, and, to a smaller extent, in the TREC DL 2019 and TREC DL 2020 datasets.

Upon comparing the performance of our models across both ranking strategies (end-to-end and re-ranking) we observe that while the end-to-end models exhibit slightly better performance, the disparity is not meaningful when considering the cost associated with each strategy. Thus, our approaches demonstrate efficacy across both ranking setups.

Finally, in comparing the fully trained model against the low-resource training approach, which solely trains the query encoder, we see that the encoder-only training not only achieves comparable performance to the fully trained model but also improves its performance in certain setups.

Therefore, in addressing the first research question, we argue that both proposed models outperform existing alternatives in keyphrase search scenarios. When considering the choice between the fully trained alternative and

---

[5]mistralai/Mistral-7B-Instruct-v0.2

Table 3
Performance of the proposed keyphrase-based models using automatically generated keyphrase queries. † (‡) mark denotes significance against the `ColBERT` re-ranking (end-to-end) baseline.

Model	TREC DL 2019			TREC DL 2020			MSMarco Dev Small			
	MAP@1k	nDCG@10	MRR@10	MAP@1k	nDCG@10	MRR@10	MRR@10	R@50	R@200	R@1k
BM25	0.2859	0.4751	0.6120	0.2857	0.4658	0.6416	0.1767	0.5785	0.7311	0.8580
BM25 ≫ ColBERT	0.4499	0.6849	0.8516	0.4475	0.6480	0.7585	0.3146	0.7572	0.8324	0.8580
BM25 ≫ ColBERTKP$_{QD}$	**0.4726**†	0.7060	0.8876	0.4579	**0.6881**†	0.8076	0.3263†	0.7685†	0.8368†	0.8580
BM25 ≫ ColBERTKP$_Q$ & ColBERT$_D$	0.4688	**0.7190**†	0.8915	**0.4609**	0.6827†	0.7898	**0.3300**†	0.7654†	0.8379†	0.8580
ColBERT	0.4303	0.6869	0.8818	0.4416	0.6348	0.7564	0.3165	0.7795	0.8744	0.9254
ColBERTKP$_{QD}$	0.4505	0.7072	0.8954	0.4558	0.6831‡	**0.8192**	0.3285‡	**0.8017**‡	**0.8939**‡	**0.9396**‡
ColBERTKP$_Q$ & ColBERT$_D$	0.4573	**0.7190**	**0.9147**	0.4508	0.6728‡	0.7970	0.3312‡	0.7977‡	0.8897‡	0.9341‡

Table 4
Performance of the proposed keyphrase-based models using the original query set. ⋄ and ∧ marks denote statistically equivalence (TOST) against `ColBERT` re-ranking and end-to-end baselines respectively.

Model	TREC DL 2019			TREC DL 2020			MSMarco Dev Small			
	MAP@1k	nDCG@10	MRR@10	MAP@1k	nDCG@10	MRR@10	MRR@10	R@50	R@200	R@1k
BM25	0.3031	0.4989	0.6780	0.2974	0.4793	0.6457	0.1850	0.5952	0.7481	0.8677
BM25 ≫ ColBERT	0.4596	0.7039	0.8353	**0.4828**	0.7146	0.8380	0.3524	0.7909	0.8523	0.8677
BM25 ≫ ColBERTKP$_{QD}$	**0.4608**	0.7107⋄	0.8469⋄	0.4797⋄	0.7096⋄	**0.8627**⋄	0.3507	0.7871	0.8524	0.8677
BM25 ≫ ColBERTKP$_Q$ & ColBERT$_D$	0.4579⋄	0.7062⋄	0.8553⋄	0.4816⋄	**0.7154**⋄	0.8356⋄	0.3556	0.7841	0.8531	0.8677
ColBERT	0.4452	0.7078	0.8574	0.4692	0.6866	0.8318	**0.3572**	**0.8221**	**0.9088**	0.9491
ColBERTKP$_{QD}$	0.4378∧	**0.7140**∧	**0.8667**∧	0.4609∧	0.6892∧	0.8576∧	0.3541	0.8179	0.9081	**0.9495**
ColBERTKP$_Q$ & ColBERT$_D$	0.4350∧	0.7075∧	0.8566∧	0.4580∧	0.6798∧	0.8602∧	0.3527	0.8116∧	0.9040	0.9462

the encoder-only alternative, the latter is preferable, providing comparable performance with fewer resources required for model training in all evaluation setups.

Observation 1

Keyphrase-tailored models outperform existing dense retrieval models in keyphrase-based search using both end-to-end and re-ranking strategies.

5.2.2. RQ2. Is the performance of keyphrase-based models degraded when using original queries?

Having demonstrated the superiority of keyphrase-tailored models in keyphrase search scenarios, we now study their performance when using original (mostly question-like) queries. In this analysis, our objective is not to achieve superior performance for the presented models but to determine their equivalence with existing approaches.

Table 4 presents the results of the previous experiment conducted with the original TREC DL and MSMarco queries. Here, we employ the TOST equivalence test to measure the equivalence between our proposed models and the baselines, as detailed in Section 5.1. Equivalence tests reveal that keyphrase-based models demonstrate comparable performance on the TREC DL datasets, except for the BM25 ≫ ColBERTKP$_{QD}$ pipeline on TREC DL 2019 in MAP@1000, where results show superiority rather than equivalence. However, due to the larger query set, equivalence is not consistently observed on the MSMarco dataset. Here, results vary, with both the proposed approaches and the baselines outperforming each other depending on specific scenarios. These results indicate that the keyphrase-based
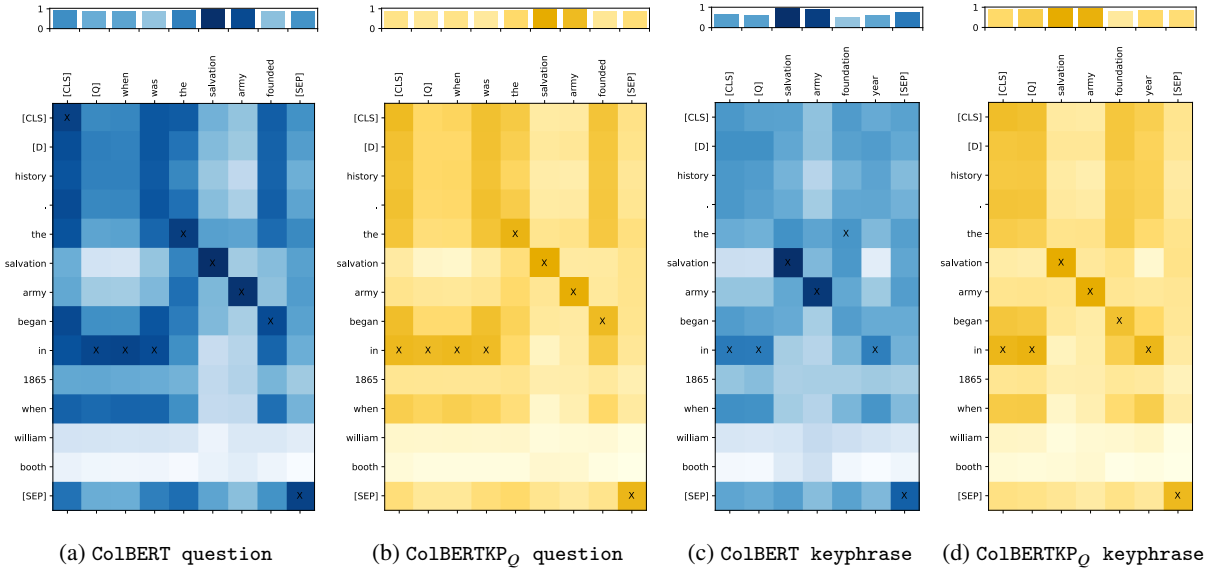

(a) ColBERT question (b) ColBERTKP$_Q$ question (c) ColBERT keyphrase (d) ColBERTKP$_Q$ keyphrase

Figure 2: ColBERT and ColBERTKP$_Q$ interaction for query 962179 (in both original and keyphrase format) and passage 2329699 (shortened). Darker shading in the interaction matrix denotes higher similarity, while the × symbol highlights the document embedding (row) with the highest similarity for each query embedding (column). The histogram above illustrates each query embedding's contribution to the documents's final score, with shading indicating the magnitude of the contribution.

models achieve comparable performance against the baselines and demonstrate improvements in specific datasets and metrics.

Observation 2

The performance of keyphrase-based models remains stable across different types of queries, showing equivalent performance to existing dense ranking models.

Continuing with *RQ1* and *RQ2*, for deeper insights into the superior performance of keyphrase-based models over baselines in keyphrase search scenarios, Figure 2 presents interaction diagrams showing the relationship between a query (on both its original and keyphrase versions) and a perfectly relevant document for the query (Table 1) for both ColBERT and ColBERTKP$_Q$ models. Each diagram shows the similarity between query and document embeddings, with darker shading in the interaction matrix indicating higher similarity scores. The × symbol represents the document embedding (rows) with the highest similarity for each query embedding (columns). Additionally, the histogram above each diagram shows the contribution of each query embedding to the document's final score.

Indeed in Table 4, we observe that the performance between ColBERT and ColBERTKP$_Q$ is nearly identical when employing original queries, resulting in similar interaction diagrams (Figures 2a and 2b) with minor discrepancies. However, that is not the case for the keyphrase-format query interaction diagrams (Figures 2c and 2d). The similarities between the keyphrase query and the document regarding the ColBERT model greatly diminish, indicating a lower final score for the perfectly relevant document. Conversely, ColBERTKP$_Q$ exhibits minimal variation from its performance with the original query, yielding high scores and consequently elevating the document's rank.

5.2.3. RQ3. Does the keyphrase-based training generalise to other models?

To study whether training models with keyphrases can be applied beyond the scope of ColBERT, we conducted an experiment that compared the performance of the monoT5 baseline from the previous experiment with a monoT5 model trained on the same keyphrase-based triples employed in training our proposed ColBERT-based models. We name this model monoT5KP.

Table 5
Performance of the `monoT5KP` re-ranking model compared to `monoT5` and `ColBERTKP`$_Q$ re-ranking pipelines. Bold values show the best performance among all setups while underscored values highlight the best results among each query set.

Model	TREC DL 2019			TREC DL 2020		
	MAP@1k	nDCG@10	MRR@10	MAP@1k	nDCG@10	MRR@10
Original Queries						
BM25 » `monoT5`	**0.4815**	**0.7238**	0.8798	**0.4957**	0.7138	**0.8704**
BM25 » `monoT5KP`	0.4597	0.7119	**0.9019**	0.4502	0.6667	0.8201
BM25 » `ColBERTKP`$_Q$	0.4579	0.7062	0.8553	0.4816	**0.7154**	0.8356
Mistral Keyphrases						
BM25 » `monoT5`	0.4628	0.6982	0.7987	0.4546	0.6801	0.8299
BM25 » `monoT5KP`	0.4574	0.7060	<u>0.8930</u>	0.4583	<u>0.6920</u>	<u>0.8448</u>
BM25 » `ColBERTKP`$_Q$	<u>0.4688</u>	<u>0.7190</u>	0.8915	<u>0.4609</u>	0.6827	0.7898

We trained the `monoT5KP` model using the script provided within the PyGaggle (Pradeep et al., 2023) library. Specifically, we train the model for 5k steps using a batch size of 64 (further training did not improve the model's effectiveness). Different from `ColBERT`-based approaches, starting from an existing `monoT5` checkpoint does not result in better performance so we report results using `t5-base` as the base model.

In Table 5, we replicate the ranking experiments from *RQ1* and *RQ2* to assess the performance of the `monoT5KP` model for both original and keyphrase queries (*Mistral Keyphrases*). For brevity, we only report results for the TREC DL datasets. Results show that the keyphrase-tailored model (`monoT5KP`) exhibits better performance than the original one (`monoT5`) when assessed with keyphrase queries across all scenarios, except for MAP@1k in the TREC DL 2019 evaluation set. In this case, the difference between both models is subtle, slightly favouring the default training approach.

To further analyse these results, we compare the performance of `monoT5KP` and `ColBERTKP`$_Q$. For the original queries, both models perform similarly, with `monoT5KP` showing a slight edge on the TREC DL 2019 dataset and `ColBERTKP`$_Q$ slightly ahead on the TREC DL 2020 dataset. When evaluating the keyphrase queries, the two approaches again demonstrate comparable performance: on the TREC DL 2020 dataset, the T5-based model outperforms the ColBERT-based model on two out of three metrics, whereas on the TREC DL 2019 dataset, the ColBERT-based model surpasses the T5-based model in two out of three metrics.

> **Observation 3**
>
> The keyphrase-based training approach exhibits versatility, as demonstrated by its successful application to models like `monoT5` whose architecture is a cross-encoder instead of late interaction. This adaptability highlights the applicability of our methodology across diverse model architectures.

5.2.4. RQ4. Does the effectiveness observed with automatically generated keyphrases extend to manually annotated ones?

To address this research question, we use the curated test query set established for TREC DL 2019, as outlined in Section 4.3. Here, we report metrics using a micro-average strategy instead of computing them individually for each assessor and subsequently averaging their scores.

This experiment aims to validate the conclusions from *RQ1* (§ 5.2.1). Here, we study if the superior performance of keyphrase-based models when employing automatically generated keyphrase queries extends to keyphrase queries crafted by human assessors.

Table 6

Results for manual keyphrases in TREC DL 2019. † (‡) mark denotes significance against the ColBERT re-ranking (end-to-end) baseline.

Model	MAP@1k	nDCG@10	MRR@10
BM25	0.2248	0.3893	0.5488
BM25 ≫ ColBERT	0.3622	0.5739	0.7093
BM25 ≫ ColBERTKP$_{QD}$	**0.3858**†	0.5970†	0.7416†
BM25 ≫ ColBERTKP$_Q$ & ColBERT$_D$	0.3816†	0.6040†	0.7566†
ColBERT	0.3477	0.5717	0.7226
ColBERTKP$_{QD}$	0.3774‡	0.6032‡	0.7633‡
ColBERTKP$_Q$ & ColBERT$_D$	0.3782‡	**0.6131**‡	**0.7829**‡

Table 7

Statistics for manually and automatically generated keyphrases for the TREC DL 2019 test queries.

	Mistral	Assessor 1	Assessor 2	Assessor 3
# keyphrases per query	1.09	2.91	1.37	1.49
# words per keyphrase	3.19	2.76	2.66	3.33
Overlap	-	0.72	0.73	0.80

Table 6 shows the results of evaluating the proposed models on the TREC DL 2019 dataset using manually annotated keyphrase queries. For this experiment, we decided to employ only the first keyphrase annotated by each assessor, given that models are usually trained to handle one keyphrase at a time. This table underscores the dominance of keyphrase-based models over existing dense retrieval models. Our keyphrase-tailored models outperform all baselines across both end-to-end and re-ranking methodologies, achieving significant improvements on all reported metrics. This further underscores the importance of developing and deploying ranking models tailored for keyphrase-based search scenarios.

Additionally, a compelling finding from this experiment is that the automatically generated keyphrases outperform those generated by humans, not only for the keyphrase-based models (which are trained using keyphrases generated in the same manner) but also for the baseline models.

In concluding this experiment, we aim to delineate the primary differences between the assessor-generated keyphrases and those generated by the LLM.

Table 7 shows some statistics about the manually labelled and automatically generated keyphrases. First, we can see that manual annotators tend to use queries with more keyphrases so that they fully cover the topic they want to search about. In fact, while the number of keyphrases per query for the automatically generated queries hovers around 1, *Assessor 1* averages nearly 3 keyphrases per query, whereas *Assessor 2* and *Assessor 3* average closer to 1.5 keyphrases per query. However, the keyphrase length (in terms of words per keyphrase) tends to be slightly shorter for the manually labelled ones.

Finally, we calculate the term overlap between automatic and manually labelled keyphrases defined as:

$$O(\mathcal{W}_{auto}, \mathcal{W}_{manual}) = \frac{|\mathcal{W}_{auto} \cap \mathcal{W}_{manual}|}{min\left(|\mathcal{W}_{auto}|, |\mathcal{W}_{manual}|\right)}$$

where $\mathcal{W}$ is the set of words for a given keyphrase query.

The results in Table 7 reveal a considerable overlap between automatically generated and manually created keyphrases of approximately 75%.
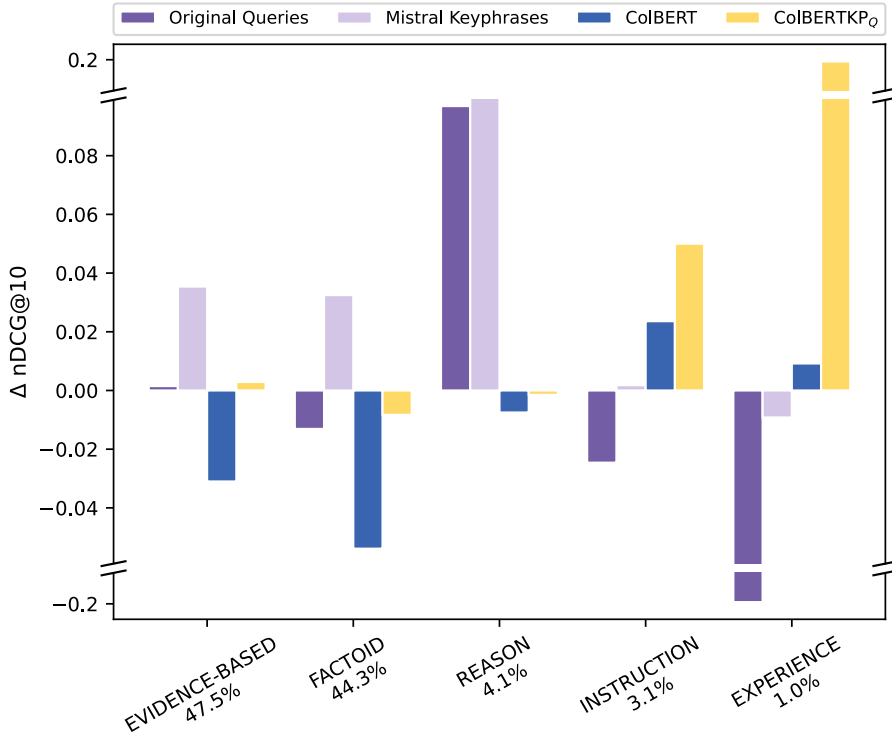
Figure 3: Performance across different types of queries according to the query type taxonomy presented by Bolotova et al. (2022). Original and Mistral keyphrases bars show the delta between using `ColBERTKP_Q` and `ColBERT` as the retrieval model. `ColBERTKP_Q` and `ColBERT` bars show the delta between using keyphrases and original queries.

Observation 4

The superior performance of keyphrase-based models using human-generated keyphrase queries reaffirms our findings when utilising automatically generated keyphrases.

5.2.5. RQ5. How are keyphrase models and queries different from original queries and models?

We want to identify the query types where each model behaves best and show differences in query-document matching behaviour between keyphrase-based and existing models.

Initially, we examine how different model types and queries perform across query types, leveraging the taxonomy proposed in (Bolotova et al., 2022) to categorise queries from the TREC DL 2019 and 2020 datasets. This classification is always done using the original version of the queries. The results of this analysis are shown in Figure 3.

In Figure 3, the *Original Queries* and *Mistral Keyphrases* bars illustrate the performance disparity between `ColBERTKP_Q` and `ColBERT`. Positive values mean better performance by the `ColBERTKP_Q` model. Here, `ColBERTKP_Q` demonstrates resilience in both query types, while `ColBERT` exhibits a substantial decline when using keyphrase-format queries, particularly in the predominant query types.

Subsequently, the `ColBERT` and `ColBERTKP_Q` bars show the performance variation between automatically generated keyphrases and original queries. Positive values mean better performance by keyphrase queries. We observe that `ColBERTKP_Q` outperforms `ColBERT` across most categories with keyphrase queries while maintaining comparable performance with original queries.

Table 8

Impact of different types of matching behaviour for TREC DL 2019 on nDCG@10, and relative decrease from All (Δ). All reported `ColBERT` results are within the end-to-end dense retrieval scenario.

Models	All Types	Lexical Match		Semantic Match		Special Match	
	nDCG@10	nDCG@10	Δ	nDCG@10	Δ	nDCG@10	Δ
Original Queries							
BM25	0.4795	-	-	-	-	-	-
ColBERT	0.7078	0.5367	-24.2%	0.0332	-95.3%	0.6820	-3.6%
ColBERTKP$_{QD}$	0.7140	0.5053	-29.2%	0.0220	-96.9%	0.6994	-2.0%
ColBERTKP$_Q$ & ColBERT$_D$	0.7075	0.5465	-22.8%	0.0375	-94.7%	0.7008	-0.9%
Mistral Keyphrases							
BM25	0.4174	-	-	-	-	-	-
ColBERT	0.6869	0.5655	-17.6%	0.0178	-97.4%	0.6774	-1.4%
ColBERTKP$_{QD}$	0.7072	0.5665	-19.9%	0.0363	-95.9%	0.7030	-0.6%
ColBERTKP$_Q$ & ColBERT$_D$	0.7190	0.5706	-20.6%	0.0383	-94.7%	0.7159	-0.4%

> **Observation 5**
>
> Keyphrase-based models consistently maintain strong performance across all categories, regardless of whether original or keyphrase-format queries are employed, while the standard model exhibits decreased effectiveness on keyphrase queries.

Our second experiment examines how training `ColBERT` on keyphrases affects its matching behaviour. Following the methodology of Wang et al. (2023), we analyse the three matching strategies between query and document. *Lexical matching* (Formal et al., 2022) refers to cases where the query and document tokens are exactly the same (e.g., *"salvation"* with *"salvation"*). In contrast, *semantic matching* (Formal et al., 2021) identifies pairs of different tokens (e.g., *"foundation"* with *"began"*), capturing semantic connections between terms. Finally, *special token matching* focuses on the alignment of `ColBERT`'s special tokens, including `[CLS]`, `[Q]`, `[SEP]`, and `[MASK]`. Using an interaction example from Figure 2, we visually represent each matching type using colours and different line styles in Figure 3.

Table 8 shows retrieval effectiveness through these three matching approaches for the TREC DL 2019 test queries. While both original and keyphrase-based models exhibit similar performance, keyphrase-based models demonstrate an advantage in semantic and special token matching, particularly evident with keyphrase-format queries. This is exemplified in Figure 2, where considerable differences emerge regarding special tokens and semantic matching between `ColBERT` and `ColBERTKP`$_Q$ with the keyphrase-like query. The latter shows higher similarities for the `[CLS]` and `[Q]` tokens and for *"foundation"* and *"began"*.
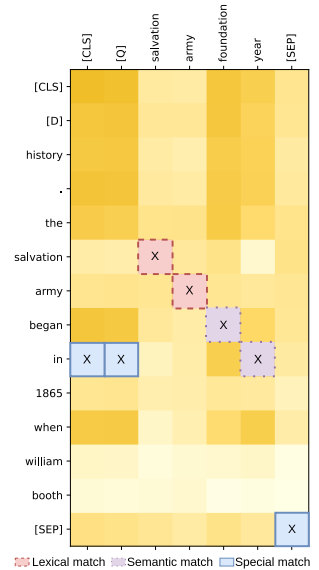

Figure 4: Representation of each matching type.

> **Observation 6**
>
> Training `ColBERT` with shorter queries enhances special token matching performance, reducing reliance on lexical matches. Token matching nearly achieves parity with the default approach encompassing all matching types.

Table 9
Results on a hybrid query format scenario using TREC DL 2019 and manually annotated queries.

Model	MAP@1k	nDCG@10	MRR@10
BM25 » ColBERT	0.4102	0.6373	0.7705
BM25 » ColBERTKP$_{QD}$	0.4224	0.6518	0.7908
BM25 » ColBERTKP$_Q$ & ColBERT$_D$	0.4192	0.6530	0.8005
ColBERT	0.4122	0.6430	0.7769
ColBERTKP$_{QD}$	**0.4246**	0.6574	0.7971
ColBERTKP$_Q$ & ColBERT$_D$	0.4213	**0.6584**	**0.8067**

5.2.6. RQ6.
How effectively do the proposed models adapt to mixed query scenarios?

Building on our analysis of how question-based and keyphrase-based models perform on their respective query types, we now explore how they perform in a mixed scenario where half of the queries are keyphrase-based and the other half are question-like queries.

Specifically, we create a synthetic dataset where 50% of the queries are question-like (from the original TREC queries), and the remaining are manually annotated keyphrase queries (§ 4.3). We compare the performance of the standard ColBERT model with the two proposed approaches (ColBERTKP$_{QD}$ and ColBERTKP$_Q$ & ColBERT$_D$).

Note that this experiment contains a random component in the query selection process. To mitigate this limitation, we ran the experiment 100 times and averaged the results. In each iteration, 21 queries of each type are randomly selected without overlap. Since three different assessors labelled the manual queries, each run randomly selects queries from one of the assessors.

Table 9 presents the retrieval effectiveness of each strategy on the mixed query set. Both ColBERTKP$_{QD}$ and ColBERTKP$_Q$ & ColBERT$_D$ demonstrate comparable performance, outperforming the standard ColBERT model in both re-ranking and end-to-end retrieval setups. When comparing the two proposed approaches, the encoder-only version (ColBERTKP$_Q$ & ColBERT$_D$) emerges as the best-performing model.

Table 9 presents the retrieval effectiveness of each strategy on the mixed query set. Both ColBERTKP$_{QD}$ and ColBERTKP$_Q$ & ColBERT$_D$ demonstrate comparable performance, outperforming the standard ColBERT model in both re-ranking and end-to-end retrieval setups. Specifically, the proposed models show gains of up to 3.01% in MAP@1k, 2.45% in nDCG@10, and 3.89% in MRR@10, depending on the setup. When comparing the two proposed approaches, the encoder-only version (ColBERTKP$_Q$ & ColBERT$_D$) emerges as the best-performing model.

> **Observation 7**
>
> The mixed query experiment results demonstrate the proposed models' superiority over the standard ColBERT approach. These findings suggest that training the model on both query types can enhance retrieval performance in mixed, real-world, query scenarios.

5.2.7. RQ7. Does the effectiveness of keyphrase-tailored models translate to traditional keyword (title-only) queries?

To conclude the experiments section, we examine whether the findings from keyphrase queries also hold for traditional title-query scenarios. We report the ranking performance of both proposed models (ColBERTKP$_{QD}$ and ColBERTKP$_Q$) on title queries from the TREC Robust 2004, TREC 7, and TREC 8 collections.

Table 10 demonstrates that the effectiveness of the keyphrase models extends to title queries. Both proposed models (ColBERTKP$_{QD}$ and ColBERTKP$_Q$) outperform the baselines across all datasets, with significant improvements observed on TREC Robust 2004 and TREC 7.

Table 10

Performance of the proposed keyphrase-based models using traditional title queries. † mark denotes significance against the `ColBERT` re-ranking baseline.

Model	TREC Robust 2004		TREC 7		TREC 8	
	MAP@1k	nDCG@10	MAP@1k	nDCG@10	MAP@1k	nDCG@10
BM25	0.2246	0.4313	0.1719	0.4537	0.2279	0.4764
BM25 » ColBERT	0.2490	0.4725	0.1955	0.5023	0.2349	0.4901
BM25 » ColBERTKP$_{QD}$	**0.2612**†	**0.4970**†	**0.2136**†	**0.5595**†	0.2417	**0.5146**
BM25 » ColBERTKP$_Q$ & ColBERT$_D$	0.2607†	0.4914†	0.2099†	0.5448†	**0.2423**	0.5032

Comparing the fully trained model (`ColBERTKP`$_{QD}$) with the encoder-only approach (`ColBERTKP`$_Q$) reveals that, while their performances are similar, the former achieves higher scores across nearly all evaluation settings, consistent with previous experiments.

In response to the eighth research question, we argue that training dense retrieval models on a keyphrase-based collection positively impacts performance even when applied to traditional title-based queries.

Observation 8

Adapting models to keyphrase-oriented data enhances their generalisability across different query formats, improving retrieval effectiveness in broader scenarios.

---

6. Conclusions

This paper introduces two keyphrase-based ranking models, namely `ColBERTKP`$_{QD}$ and `ColBERTKP`$_Q$, designed to enhance the performance of dense retrieval systems in keyphrase search scenarios. The former encompasses training the full model architecture, while the latter focuses solely on training the query encoder, reducing training resource requirements. Our experimental setup includes the generation of keyphrase-based datasets for training and testing, adapting original TREC and MSMarco queries using an LLM. Furthermore, we manually label the TREC DL 2019 dataset test queries, providing human-labelled data for performance assessment.

Demonstrating the generalisability of our training approach, we trained a text-to-text re-ranking model (`monoT5KP`) with promising results. Additionally, we investigate the performance of our models across various query categories, revealing their robustness and effectiveness across different query types.

Our detailed analysis of matching strategies reveals important insights into the behaviour of keyphrase-based models. Training `ColBERT` with keyphrases enhanced special token matching performance and reduced dependence on lexical matches, demonstrating the adaptability of our approach.

Finally, we demonstrate the adaptability of our approach to other query types, specifically using traditional title-based queries from classic IR collections. Additionally, we conduct tests on hybrid query scenarios containing both keyphrase-like and question-like queries, showing our models' superior performance across these varied formats.

In future work, we plan on testing `ColBERTv2` (Santhanam et al., 2022) for keyphrase search. Additionally, applying distillation techniques to refine these models could facilitate more efficient knowledge transfer and model compression without compromising performance. Moreover, creating query and document pruning strategies using keyphrase information could substantially improve the efficiency of the proposed retrieval strategies. Another research involves developing a ranking pipeline that first classifies a query as a question or keyphrase and then selects the appropriate model to encode the query. Lastly, we aim to explore the application of these models to boolean queries, which use keyphrases as terms joined by boolean operators.

Table 11
Examples of automatically generated and manually annotated keyphrases for TREC DL 2019 queries.

Query	Method	Generated/Annotated Keyphrases
do goldfish grow	**T5 AKG**	goldfish, squid
	Mistral	goldfish growth
	Annotator 1	goldfish growth, goldfish size, goldfish max size
	Annotator 2	goldfish size, goldfish max size
	Annotator 3	goldfish growth
what is wifi vs bluetooth	**T5 AKG**	csm, wifi, bluetooth
	Mistral	wifi vs bluetooth
	Annotator 1	wifi bluetooth advantages, wifi versus bluetooth, wifi vs bluetooth
	Annotator 2	wifi vs bluetooth
	Annotator 3	wifi vs bluetooth, wifi bluetooth differences
difference between rn and bsn	**T5 AKG**	bsa, rn-bsn, rn
	Mistral	RN vs BSN, registered nurse vs bachelor of science in nursing
	Annotator 1	bachelor science nursing vs registered nurse, bsn vs rn
	Annotator 2	rn vs bsn, bsn rs difference
	Annotator 3	rn vs bsn, rn bsn differences, registered nurse bachelor of science in nursing differences, registered nurse vs bachelor of science in nursing
rsa definition key	**T5 AKG**	rsa definition, rsa
	Mistral	RSA definition key
	Annotator 1	rsa key, rsa key definition, Rivest Shamir Adleman key
	Annotator 2	rsa definition key
	Annotator 3	rsa key, rsa key definition
hydrogen is a liquid below what temperature	**T5 AKG**	hydrogen, hydrogen atom
	Mistral	hydrogen liquid temperature
	Annotator 1	hydrogen condensation temperature, liquid hydrogen temperature
	Annotator 2	hydrogen temperature
	Annotator 3	hydrogen melting temperature
why did the us voluntarily enter ww1	**T5 AKG**	ww1
	Mistral	us volunteer entry ww1 reason
	Annotator 1	united states word war 1, usa ww1, usa enters ww1, united stated world war I, usa wwI, usa enters wwI
	Annotator 2	us ww1 incentives, us ww1 participation
	Annotator 3	us ww1 voluntary participation causes, us world war 1 voluntary participation causes, united states world war 1 voluntary participation causes, united states ww1 voluntary participation causes

A. Questions to Keyphrases Examples

Table 11 shows examples of translating original TREC DL 2019 queries into keyphrase format. We report keyphrases generated from three methods: T5 AKG, Mistral, and manual annotations provided by three annotators.

B. Prompt Template

Listing 1: Mistral prompt to generate keyphrase-format queries.

```
1   [INST]
2   You are an expert in transforming a query into keyphrase format.
3
4   You must follow these rules:
5   1. When the query is a keyphrase you must just return the query and nothing else (examples of keyphrases
        ↪ are: information retrieval, solar panels, quantum computing, heart rate or blockchain).
6   2. Try not to include variations of the same keyphrase.
7   3. Acronyms are also considered keyphrases.
8   4. One-word keyphrases are allowed.
9   5. Do not provide several sets of keyphrases stick to the output format (do not add "Or: {{<Keyphrases
        ↪ >{{keyphrase 1}}, {{keyphrase 2}}</Keyphrases>}}")
10
11  The input format is:
12  Generate keyphrases for:
13  <Query>{{input query}}</Query>
14
15  The output format is:
16  <Keyphrases>{{keyphrase 1}}, {{keyphrase 2}}</Keyphrases>
17
18  Generate keyphrases for:
19  <Query>how many calories in jiffy natural peanut butter</Query>
20  [/INST]
21  <Keyphrases>jiffy natural peanut butter calories</Keyphrases>
22  </s>
23
24  <s>[INST]
25  Generate keyphrases for:
26  <Query>where do belgian draft horses originate</Query>
27  [/INST]
28  <Keyphrases>belgian draft horses origin</Keyphrases>
29  </s>
30
31  <s>[INST]
32  Generate keyphrases for:
33  <Query>who are the royal guards</Query>
34  [/INST]
35  <Keyphrases>royal guard</Keyphrases>
36  </s>
37
38  <s>[INST]
39  Generate keyphrases for:
40  <Query>{query}</Query>
41  [/INST]
```

CRediT authorship contribution statement

Jorge Gabín: Conzeptualization, Software, Investigation, Writing - Original Draft, Visualization. **Javier Parapar:** Conzeptualization, Writing - Review & Editing, Supervision. **Craig Macdonald:** Conzeptualization, Writing - Review & Editing, Supervision.

Acknowledgements

References

J. Dalton, S. Fischer, P. Owoicho, F. Radlinski, F. Rossetto, J. R. Trippas, H. Zamani, Conversational information seeking: Theory and application, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 3455–3458. URL: https://doi.org/10.1145/3477495.3532678.

K. GUO, C. Defretiere, D. Diefenbach, C. Gravier, A. Gourru, Qanswer: Towards question answering search over websites, in: Companion Proceedings of the Web Conference 2022, WWW '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 252–255. URL: https://doi.org/10.1145/3487553.3524250.

D. Khurana, A. Koli, K. Khatter, S. Singh, Natural language processing: state of the art, current trends and challenges, Multimedia Tools and Applications 82 (2023) 3713–3744.

P. Jacsó, Academic search engines: A quantitative outlook, Online Inf. Rev. 39 (2015) 435–436.

X. Li, B. J. Schijvenaars, M. de Rijke, Investigating queries and search failures in academic search, Information Processing & Management 53 (2017) 666–683.

T. Russell-Rose, J. Chamberlain, L. Azzopardi, Information retrieval in the workplace: A comparison of professional search practices, Information Processing & Management 54 (2018) 1042–1057.

A. Lahiri, D. K. Sanyal, I. Mukherjee, A keyphrase-centric search engine for scientific papers, in: Proceedings of the 15th Annual Meeting of the Forum for Information Retrieval Evaluation, FIRE '23, Association for Computing Machinery, New York, NY, USA, 2024, p. 125–128. URL: https://doi.org/10.1145/3632754.3632772.

T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, L. Deng, MS MARCO: A human generated machine reading comprehension dataset, in: T. R. Besold, A. Bordes, A. S. d'Avila Garcez, G. Wayne (Eds.), Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016, volume 1773 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2016. URL: https://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), Association for Computational Linguistics, 2019, pp. 4171–4186. URL: https://doi.org/10.18653/v1/n19-1423.

V. Karpukhin, B. Oguz, S. Min, P. S. H. Lewis, L. Wu, S. Edunov, D. Chen, W. Yih, Dense passage retrieval for open-domain question answering, in: B. Webber, T. Cohn, Y. He, Y. Liu (Eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, Association for Computational Linguistics, 2020, pp. 6769–6781. URL: https://doi.org/10.18653/v1/2020.emnlp-main.550.

L. Xiong, C. Xiong, Y. Li, K. Tang, J. Liu, P. N. Bennett, J. Ahmed, A. Overwijk, Approximate nearest neighbor negative contrastive learning for dense text retrieval, in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, OpenReview.net, 2021. URL: https://openreview.net/forum?id=zeFrfgyZln.

O. Khattab, M. Zaharia, Colbert: Efficient and effective passage search via contextualized late interaction over BERT, in: J. X. Huang, Y. Chang, X. Cheng, J. Kamps, V. Murdock, J. Wen, Y. Liu (Eds.), Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020, ACM, 2020, pp. 39–48. URL: https://doi.org/10.1145/3397271.3401075.

X. Wang, C. Macdonald, N. Tonellotto, I. Ounis, Reproducibility, replicability, and insights into dense multi-representation retrieval models: from colbert to col, in: H. Chen, W. E. Duh, H. Huang, M. P. Kato, J. Mothe, B. Poblete (Eds.), Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023, ACM, 2023, pp. 2552–2561. URL: https://doi.org/10.1145/3539618.3591916.

J. Ni, C. Qu, J. Lu, Z. Dai, G. H. Ábrego, J. Ma, V. Y. Zhao, Y. Luan, K. B. Hall, M. Chang, Y. Yang, Large dual encoders are generalizable retrievers, in: Y. Goldberg, Z. Kozareva, Y. Zhang (Eds.), Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022, Association for Computational Linguistics, 2022, pp. 9844–9855. URL: https://doi.org/10.18653/v1/2022.emnlp-main.669.

J. Lee, Z. Dai, S. M. K. Duddu, T. Lei, I. Naim, M. Chang, V. Zhao, Rethinking the role of token retrieval in multi-vector retrieval, in: A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, S. Levine (Eds.), Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023, 2023. URL: http://papers.nips.cc/paper_files/paper/2023/hash/31d997278ee9069d6721bc194174bb4c-Abstract-Conference.html.

M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P. Mazaré, M. Lomeli, L. Hosseini, H. Jégou, The faiss library, CoRR abs/2401.08281 (2024).

J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with gpus, IEEE Trans. Big Data 7 (2021) 535–547.

C. Silverstein, M. Henzinger, H. Marais, M. Moricz, Analysis of a very large web search engine query log, volume 33, 1999, pp. 6–12. URL: https://doi.org/10.1145/331403.331405. doi:10.1145/331403.331405.

J. H. Reimer, S. Schmidt, M. Fröbe, L. Gienapp, H. Scells, B. Stein, M. Hagen, M. Potthast, The archive query log: Mining millions of search result pages of hundreds of search engines from 25 years of web archives, in: H. Chen, W. E. Duh, H. Huang, M. P. Kato, J. Mothe, B. Poblete (Eds.), Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023, ACM, 2023, pp. 2848–2860. URL: https://doi.org/10.1145/3539618.3591890.

M. Taghavi, A. Patel, N. Schmidt, C. Wills, Y. Tew, An analysis of web proxy logs with query distribution pattern approach for search engines, Computer Standards & Interfaces 34 (2012) 162–170.

V. Bolotova, V. Blinov, F. Scholer, W. B. Croft, M. Sanderson, A non-factoid question-answering taxonomy, in: E. Amigó, P. Castells, J. Gonzalo, B. Carterette, J. S. Culpepper, G. Kazai (Eds.), SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022, ACM, 2022, pp. 1196–1207. URL: https://doi.org/10.1145/3477495.3531926.

R. F. Nogueira, Z. Jiang, R. Pradeep, J. Lin, Document ranking with a pretrained sequence-to-sequence model, in: T. Cohn, Y. He, Y. Liu (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020, volume EMNLP 2020 of *Findings of ACL*, Association for Computational Linguistics, 2020, pp. 708–718. URL: https://doi.org/10.18653/v1/2020.findings-emnlp.63. doi:10.18653/V1/2020.FINDINGS-EMNLP.63.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, J. Mach. Learn. Res. 21 (2020) 140:1–140:67.

R. Pradeep, R. F. Nogueira, J. Lin, The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models, CoRR abs/2101.05667 (2021).

C. Macdonald, N. Tonellotto, I. Ounis, On single and multiple representations in dense passage retrieval, in: V. W. Anelli, T. D. Noia, N. Ferro, F. Narducci (Eds.), Proceedings of the 11th Italian Information Retrieval Workshop 2021, Bari, Italy, September 13-15, 2021, volume 2947 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021. URL: https://ceur-ws.org/Vol-2947/paper5.pdf.

S. Lin, J. Yang, J. Lin, In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval, in: A. Rogers, I. Calixto, I. Vulic, N. Saphra, N. Kassner, O. Camburu, T. Bansal, V. Shwartz (Eds.), Proceedings of the 6th Workshop on Representation Learning for NLP, RepL4NLP@ACL-IJCNLP 2021, Online, August 6, 2021, Association for Computational Linguistics, 2021, pp. 163–173. URL: https://doi.org/10.18653/v1/2021.repl4nlp-1.17.

Y. Kim, Applications and future of dense retrieval in industry, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 3373–3374. URL: https://doi.org/10.1145/3477495.3536324.

B. Clavié, Jacolbertv2. 5: Optimising multi-vector retrievers to create state-of-the-art japanese retrievers with constrained resources, arXiv preprint arXiv:2407.20750 (2024).

S. Lee, A. Shakir, D. Koenig, J. Lipp, Colbertus maximus - introducing mxbai-colbert-large-v1, 2024. URL: https://www.mixedbread.ai/blog/mxbai-colbert-large-v1, accessed: 2024-10-16.

R. Jha, B. Wang, M. Günther, S. Sturua, M. K. Akram, H. Xiao, Jina-colbert-v2: A general-purpose multilingual late interaction retriever, arXiv preprint arXiv:2408.16672 (2024).

Intel, Colbert-nq, https://huggingface.co/Intel/ColBERT-NQ, 2024. Accessed: 2024-10-16.

S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, A. Hanbury, Improving efficient neural ranking models with cross-architecture knowledge distillation, CoRR abs/2010.02666 (2020).

S. Hofstätter, S. Lin, J. Yang, J. Lin, A. Hanbury, Efficiently teaching an effective dense retriever with balanced topic aware sampling, in: F. Diaz, C. Shah, T. Suel, P. Castells, R. Jones, T. Sakai (Eds.), SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021, ACM, 2021, pp. 113–122. URL: https://doi.org/10.1145/3404835.3462891.

S. Lin, J. Yang, J. Lin, Distilling dense representations for ranking using tightly-coupled teachers, CoRR abs/2010.11386 (2020).

X. Wang, S. MacAvaney, C. Macdonald, I. Ounis, An inspection of the reproducibility and replicability of tct-colbert, in: E. Amigó, P. Castells, J. Gonzalo, B. Carterette, J. S. Culpepper, G. Kazai (Eds.), SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022, ACM, 2022, pp. 2790–2800. URL: https://doi.org/10.1145/3477495.3531721.

J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, S. Ma, Optimizing dense retrieval model training with hard negatives, in: F. Diaz, C. Shah, T. Suel, P. Castells, R. Jones, T. Sakai (Eds.), SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021, ACM, 2021, pp. 1503–1512. URL: https://doi.org/10.1145/3404835.3462880.

C. Lassance, M. Maachou, J. Park, S. Clinchant, Learned token pruning in contextualized late interaction over BERT (colbert), in: E. Amigó, P. Castells, J. Gonzalo, B. Carterette, J. S. Culpepper, G. Kazai (Eds.), SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022, ACM, 2022, pp. 2232–2236. URL: https://doi.org/10.1145/3477495.3531835.

K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, M. Zaharia, Colbertv2: Effective and efficient retrieval via lightweight late interaction, in: M. Carpuat, M. de Marneffe, I. V. M. Ruíz (Eds.), Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022, Association for Computational Linguistics, 2022, pp. 3715–3734. URL: https://doi.org/10.18653/v1/2022.naacl-main.272.

M. Kulkarni, D. Mahata, R. Arora, R. Bhowmik, Learning rich representation of keyphrases from text, in: M. Carpuat, M. de Marneffe, I. V. M. Ruíz (Eds.), Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022, Association for Computational Linguistics, 2022, pp. 891–906. URL: https://doi.org/10.18653/v1/2022.findings-naacl.67.

J. Gabín, M. E. Ares, J. Parapar, Exploring models for automatic keyword labelling of scientific documents, in: L. Tamine, E. Amigó, J. Mothe (Eds.), Proceedings of the 2nd Joint Conference of the Information Retrieval Communities in Europe (CIRCLE 2022), Samatan, Gers, France, July 4-7, 2022, volume 3178 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022. URL: https://ceur-ws.org/Vol-3178/CIRCLE_2022_paper_04.pdf.

M. Song, H. Jiang, S. Shi, S. Yao, S. Lu, Y. Feng, H. Liu, L. Jing, Is chatgpt A good keyphrase generator? A preliminary study, CoRR abs/2303.13001 (2023).

R. Martínez-Cruz, A. J. López-López, J. Portela, Chatgpt vs state-of-the-art models: A benchmarking study in keyphrase generation task, CoRR abs/2304.14177 (2023).

K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, M. Jaggi, Simple unsupervised keyphrase extraction using sentence embeddings, in: A. Korhonen, I. Titov (Eds.), Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018, Brussels, Belgium, October 31 - November 1, 2018, Association for Computational Linguistics, 2018, pp. 221–229. URL: https://doi.org/10.18653/v1/k18-1022.

I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, C. G. Nevill-Manning, Kea: Practical automatic keyphrase extraction, in: Proceedings of the fourth ACM conference on Digital libraries, 1999, pp. 254–255.

X. Wang, C. MacDonald, N. Tonellotto, I. Ounis, Colbert-prf: Semantic pseudo-relevance feedback for dense passage and document retrieval, ACM Trans. Web 17 (2023a) 3:1–3:39.

X. Wang, S. MacAvaney, C. Macdonald, I. Ounis, Effective contrastive weighting for dense query expansion, in: A. Rogers, J. L. Boyd-Graber, N. Okazaki (Eds.), Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023, Association for Computational Linguistics, 2023b, pp. 12688–12704. URL: https://doi.org/10.18653/v1/2023.acl-long.710.

R. Jagerman, H. Zhuang, Z. Qin, X. Wang, M. Bendersky, Query expansion by prompting large language models, CoRR abs/2305.03653 (2023).

L. Wang, N. Yang, F. Wei, Query2doc: Query expansion with large language models, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023, Association for Computational Linguistics, 2023, pp. 9414–9423. URL: https://aclanthology.org/2023.emnlp-main.585.

M. Alaofi, L. Gallagher, M. Sanderson, F. Scholer, P. Thomas, Can generative llms create query variants for test collections? an exploratory study, in: H. Chen, W. E. Duh, H. Huang, M. P. Kato, J. Mothe, B. Poblete (Eds.), Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023, ACM, 2023, pp. 1869–1873. URL: https://doi.org/10.1145/3539618.3591960.

R. Nogueira, J. Lin, A. Epistemic, From doc2query to docttttttquery, Online preprint 6 (2019).

B. B. Cambazoglu, L. Tavakoli, F. Scholer, M. Sanderson, W. B. Croft, An intent taxonomy for questions asked in web search, in: F. Scholer, P. Thomas, D. Elsweiler, H. Joho, N. Kando, C. Smith (Eds.), CHIIR '21: ACM SIGIR Conference on Human Information Interaction and Retrieval, Canberra, ACT, Australia, March 14-19, 2021, ACM, 2021, pp. 85–94. URL: https://doi.org/10.1145/3406522.3446027.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de Las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, W. E. Sayed, Mistral 7b, CoRR abs/2310.06825 (2023).

E. M. Voorhees, The trec robust retrieval track, SIGIR Forum 39 (2005) 11–20.

D. Hawking, N. Craswell, P. B. Thistlewaite, Overview of TREC-7 very large collection track, in: E. M. Voorhees, D. K. Harman (Eds.), Proceedings of The Seventh Text REtrieval Conference, TREC 1998, Gaithersburg, Maryland, USA, November 9-11, 1998, volume 500-242 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 1998, pp. 40–52.

D. Hawking, E. M. Voorhees, N. Craswell, P. Bailey, Overview of the TREC-8 web track, in: E. M. Voorhees, D. K. Harman (Eds.), Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999, volume 500-246 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 1999. URL: http://trec.nist.gov/pubs/trec8/papers/web_overview.pdf.

N. Craswell, B. Mitra, E. Yilmaz, D. Campos, E. M. Voorhees, Overview of the TREC 2019 deep learning track, CoRR abs/2003.07820 (2020a).

N. Craswell, B. Mitra, E. Yilmaz, D. Campos, Overview of the TREC 2020 deep learning track, in: E. M. Voorhees, A. Ellis (Eds.), Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC 2020, Virtual Event [Gaithersburg, Maryland, USA], November 16-20, 2020, volume 1266 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 2020b. URL: https://trec.nist.gov/pubs/trec29/papers/OVERVIEW.DL.pdf.

C. Macdonald, N. Tonellotto, Declarative experimentation in information retrieval using pyterrier, in: K. Balog, V. Setty, C. Lioma, Y. Liu, M. Zhang, K. Berberich (Eds.), ICTIR '20: The 2020 ACM SIGIR International Conference on the Theory of Information Retrieval, Virtual Event, Norway, September 14-17, 2020, ACM, 2020, pp. 161–168. URL: https://doi.org/10.1145/3409256.3409829.

D. J. Schuirmann, A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability, Journal of pharmacokinetics and biopharmaceutics 15 (1987) 657–680.

T. Dettmers, L. Zettlemoyer, The case for 4-bit precision: k-bit inference scaling laws, in: A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, J. Scarlett (Eds.), International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA, volume 202 of *Proceedings of Machine Learning Research*, PMLR, 2023, pp. 7750–7774. URL: https://proceedings.mlr.press/v202/dettmers23a.html.

R. Pradeep, H. Chen, L. Gu, M. S. Tamber, J. Lin, Pygaggle: A gaggle of resources for open-domain question answering, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval - 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2-6, 2023, Proceedings, Part III, volume 13982 of *Lecture Notes in Computer Science*, Springer, 2023, pp. 148–162. URL: https://doi.org/10.1007/978-3-031-28241-6_10.

T. Formal, B. Piwowarski, S. Clinchant, Match your words! A study of lexical matching in neural information retrieval, in: M. Hagen, S. Verberne, C. Macdonald, C. Seifert, K. Balog, K. Nørvåg, V. Setty (Eds.), Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part II, volume 13186 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 120–127.

T. Formal, B. Piwowarski, S. Clinchant, A white box analysis of colbert, in: Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43, Springer, 2021, pp. 257–263.