Neural Pseudo-Relevance Feedback Models for Information Retrieval

Xiao Wang

Submitted in fulfilment of the requirements for the
Degree of Doctor of Philosophy

School of Computing Science
College of Science and Engineering
University of Glasgow

September 2023

Abstract

Verbatim queries submitted to search engines often do not sufficiently describe the user's search intent. Moreover, even with well-formed user queries, retrieval failures can still occur, caused by lexical or semantic mismatches, or both, between the language of the user's query and that used in the relevant documents. Pseudo-relevance feedback (PRF) techniques, which modify a query's representation using top-ranked documents, have been shown to overcome such inadequacies and improve retrieval effectiveness.

In this thesis, we argue that the pseudo-relevance feedback information can be used in neural-based models to improve retrieval effectiveness, for both sparse retrieval and dense retrieval paradigms. Indeed, recent advancements in pretrained generative language models, such as T5 and FlanT5, have demonstrated their ability to generate textual responses that are relevant to a given prompt. In light of this success, we study the capacity of such models to perform query reformulation and how they compare with long-standing query reformulation methods that use pseudo-relevance feedback. In particular, we investigate two representative query reformulation frameworks, GenQR and GenPRF. Specifically, GenQR directly reformulates the user's input query, while GenPRF provides additional context for the query by making use of pseudo-relevance feedback information in the top-ranked documents. For each reformulation method, we leverage different techniques, including fine-tuning and direct prompting, to harness the knowledge of language models. The reformulated queries produced by the generative models are demonstrated to markedly benefit the effectiveness of sparse retrieval on various TREC test collections.

In addition, Dense retrieval models, in both single representation dense retrieval and multiple representation dense retrieval paradigms, have shown higher effectiveness over traditional sparse retrieval by mitigating the lexical and semantic mismatch issues to some extent. However, underrepresented queries can still cause retrieval failures. In particular, in this thesis, we investigate the potential for multiple representation dense retrieval (exemplified by ColBERT) to be enhanced using pseudo-relevance feedback, and thereby present our proposed approach, ColBERT-PRF. More specifically, ColBERT-PRF extracts representative feedback embeddings from the document embeddings of the pseudo-relevant set and uses corresponding token statistics to identify good expansion embeddings among the representative embeddings. These expansion embeddings are then appended to the original query representation to form a refined query representation. We show that these additional expansion embeddings benefit the effectiveness of a reranking of the initial query results as well as an additional dense retrieval operation. Evaluation experiments conducted on MSMARCO passage and document ranking as well as the TREC Robust04 document ranking tasks demonstrate the effectiveness of our proposed ColBERT-PRF technique. In addition, we study the effectiveness of variants of the ColBERT-PRF model with different weighting methods. Finally, we show that ColBERT-PRF can be made more efficient, and with little impact on effectiveness, through the application of approximate scoring and different clustering methods.

While PRF techniques are effective in closing the vocabulary gap between the user's query formulations and the relevant documents, they are typically applied on the same *target* corpus as the final retrieval. In the past, external expansion techniques have sometimes been applied to obtain a high-quality pseudo-relevant feedback set using a high quality *external* corpus. However, such external expansion approaches have only been studied for sparse retrieval, and their effectiveness for recent dense retrieval methods remains under investigation. Moreover, dense retrieval approaches such as ANCE and ColBERT have been shown to face challenges when it comes to out-of-domain evaluations, due to the knowledge shift between different domains. Therefore, in this thesis, we propose a dense external expansion technique to improve the zero-shot retrieval effectiveness of both single and multiple representation dense retrieval. In particular, we employ the MSMARCO passage collection as the external corpus. The experimental results performed on two TREC datasets indicate the effectiveness of our proposed external dense query expansion techniques for both the sparse retrieval and the (single or multiple) dense retrievals.

Furthermore, we note that the ColBERT model has only been applied to the BERT model with its corresponding WordPiece tokeniser. However, the effect of the pre-trained model and the tokenisation method for the contextualised late interaction mechanism used by ColERT is not well understood. Therefore, in this thesis, we extend ColBERT to Col⋆ and ColBERT-PRF to Col⋆-PRF, by generalising the de-facto standard BERT PLM to various different PLMs. As different tokenisation methods can directly impact the matching behaviour within the late interaction mechanism, we study the nature of matches occurring in different Col⋆ and Col⋆-PRF models, and further quantify the contribution of lexical and semantic matching on retrieval effectiveness.

Finally, both the ColBERT-PRF as well as the Col⋆-PRF models perform dense query expansion in an unsupervised manner and might be affected by heuristic techniques such as clustering and IDF statistics. Therefore, in this thesis, we propose a contrastive solution that learns to select the most useful embeddings for expansion. More specifically, a deep language model-based contrastive weighting model, called CWPRF, is trained to learn to discriminate between relevant and non-relevant documents for semantic search. Our experimental results show that our contrastive weighting model can aid in selecting useful expansion embeddings and outperform various baselines. In particular, CWPRF can further improve nDCG@10 by upto 4.1% compared to our proposed ColBERT-PRF approach while maintaining its efficiency.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my principal supervisor, Prof. Craig Macdonald, and second supervisor, Prof. Iadh Ounis, for their invaluable guidance, and support for my research. Their expertise and dedication in the field of Information Retrieval have served as a continual source of motivation for me. Their insights and feedback are critical to shaping this thesis. Indeed, without their mentorship, this thesis would not have been possible. A significant portion of the work for this PhD thesis was completed during the pandemic. Beyond the guidance of research, I am also profoundly grateful to my supervisors for their unconditional encouragement and assistance in various aspects of my life, both during and after the pandemic.

I would also like to thank my outstanding collaborators, Dr. Sean MacAvaney and Prof. Nicola Tonellotto from the University of Pisa. I have not only learned numerous excellent working habits from them but also benefited immensely from their insightful perspectives and wisdom. I cherish the time we have spent together and have greatly enjoyed the insightful discussion with them.

I am also thankful to all my colleagues from the Terrier Team for their courteous and punctual encouragement throughout the course of my PhD. My gratitude extends to Erlend Frayling, Javier Sanz-Cruzado Puig, Yaxiong Wu, Thomas Jänich, Hitarth Narvala, Graham McDonald, Zaiqiao Meng, Zixuan Yi, Zeyuan Meng, Zeyan Liang, Jake Laver, Andrew Parry, Debasis Ganguly, Jack Mckechnie, Jinyuan Fang, Lubingzhi Guo, Richard McCreadie, Sarawoot Kongyoung, Sasha Petrov, Maria Vlachou, Zijun Long, Alexander Pugantsov, Andreas Chari, Ed Richard, Xi Wang, Siwei Liu and Ting Su. We had a great time in the past four years.

Last but not least, I am grateful to my family for their unconditional encouragement. I am especially grateful to my boyfriend for his meticulous support in every possible way. Lastly, I would like to extend my gratitude to my pet, Elmo, for his invaluable role as an emotional support cat throughout my PhD journey.

Contents

List of Tables

List of Figures

List of Symbols

The following list describes the symbols that will be later used within the body of the document.

q	A query
d	A document
p	A passage
p^{+}	A positive or judged relevant passage to a query
p^{-}	A negative or judged non-relevant passage to a query
Q	A query set
C	A document collection
D	A document set
N	The number of documents in the collection
D_r	A known relevant set of documents
D_n	A known non-relevant set of documents
$\mathcal{D}_{\mathcal{B}}$	A batch of documents
w	A weight
ws	A predicted weight generated by CWPRF model
θ	The parameters of a model
μ	The smoothing parameter used in the RM3 query expansion model
α	A hyperparameter which controls the contribution of the expansion terms with respect to the original query terms in the RM3 query expansion model

t	A term
$\mathbb{R}$	Real numbers
$tf_{t,d}$	The number of occurrences of the term t in the document d
$tf(t,C)$	The number of occurrences of the term t in the collection C
$tf_{avg}(t,C)$	The average number of occurrences of the term t in the collection C used in the Bo1 query expansion model
df_t	The number of documents containing a term t
idf_t	The inverse document frequency of the term t
Q_E	A query encoder
D_E	A document encoder
ϕ_q	A list of token-level embeddings for query q in multipe representation dense retrieval
ϕ_d	A list of token-level embeddings for document d in multipe representation dense retrieval
ϕ_p	A list of token-level embeddings for the pseudo-relevance feedback passage p in multiple representation dense retrieval
ϕ_{q_i}	An embedding for a query token q_i or a document token d_i
ϕ_{d_i}	An embedding for a document token d_i
ϕ_{p_i}	An embedding for a pseudo-relevance feedback token p_i
ψ_q	The embedding for query q as whole in single representation dense retrieval
ψ_d	The embedding for document d as whole in single representation dense retrieval
$s(q,d)$	The estimated relevance score for the query and document pair
$sim(\cdot)$	The similarity function
$MaxSim(\cdot)$	The maximum similarity scoring method
$log(\cdot)$	The logarithm function
$exp(\cdot)$	The exponential function
$\mathcal{L}(\cdot)$	A loss function
$\texttt{AvgPool}(\cdot)$	A function taking the average pooling on a list of token-level representations

$KL(\cdot)$ A Kullback-Leibler (KL) divergence function applied on two distributions

τ A temperature hyperparameter for the transformer-based language model

V_q A query representation

V_d A document representation

$\vec{q}_j$ A vector representation a a query term

$\vec{d}_j$ A vector representation a a document term

$f(t)$ A score function calculated based on the overlapping terms between document d and query q

$\eta(q,t)$ A query representation of the query q using the associated statistics, such as term frequency tf and document frequency df

$\eta(d,t)$ A document representation of the document d using the associated statistics, such as term frequency tf and document frequency df

» PyTerrier operator: pass the output from one transformer function to the next transformer function

$\overset{x}{\gg}$ Passing the output of one process, with type x, as input to another, where $x = R$ indicates the type is a ranking of documents and $x = Q$ indicates the type is a query

k Rank cutoff value to short a retrieved result list

% PyTerrier operator: Shorted a retrieved result list to the first k elements

I An index of a data collection

I_{target} A target index

I_{ext} An external index

$Ret(I,k)(q) \rightarrow R$ a retrieval process that takes the query q as input and returns a ranking list of k documents as the retrieved set R by searching over the index I

$Pipe$ A retrieval pipeline established for PyTerrier

$Sliding(I)$ A sliding function which applies to a text and splits it into shorter texts

$\mathcal{P}(\cdot)$ A query reformulation process to generate query reformulation(s) (Equation (2.24))

$\mathcal{P}_{QR}(\cdot)$ A query reformulation process that takes the original query text q^0 as well as its pseudo-relevance feedback documents (Equation (2.25))

$\mathcal{P}_{PRF}(\cdot)$ A query reformulation process only rely on the text of the original query

$\texttt{Retriever}(I,k')$ A retrieval model which is employed to provide a list of documents by search over the index I and the top-k' returned documents are assumed to be relevant to the search query

$\texttt{Reranker}(I,k)$ A reranking model which is employed to reorder a list of documents in accordance with their estimated relevance to the search query by searching over the index I and returns the top k returned documents to the user

$\texttt{PRF}(I,\theta)$ A pseudo-relevance technique with parameters θ and applies on the index I

$Rel(q)$ The documents that can satisfy the user's information need are judged as relevant for a query q

$P(q,k)$ The fraction of the retrieved documents that are relevant to the input query with k cutoff for a query q, calculated using Equation (2.38)

$AP(q,k)$ The average precision for a query q with cutoff k, calculated using Equation (2.5.2)

$MAP(Q,k)$ The mean average precision for a query set Q with cutoff k,calculated using Equation (2.5.2)

$R(q,k)$ The proportion of relevant documents retrieved among the total number of relevant documents wih k cutoff in the collection for a query q, calculated using Equation (2.39)

$DCG(q,k)$ The discounted cumulative gain metric for a query q and with cutoff value k, calculated using Equation (2.5.2)

$nDCG(q,k)$ The normalised discounted cumulative gain metric for a query q and with cutoff value k, calculated using Equation (2.5.2)

$nDCG(q,k)$ The ideally perfect ranking of a query q with cutoff value k, calculated using Equation (2.5.2)

$MRR(q,k)$ The reciprocal of the rank of the first relevant document retrieved of a query q with cutoff value k, calculated using Equation (2.5.2)

q^0 An original user query

q^r A reformulated query

$\langle q_x, q_y \rangle$ A query pair that q_x and q_y share at least one relevant document from a test collection

$R_K(q_x)$ Top-K result list response to a query q_x

$R_K(q_y)$ Top-K result list response to a query q_y

$\mathcal{M}$ Initial pool of query pairs for fine-tuning T5 model

$\mathcal{W}_O$ Proposed Overlap filter (Equation (3.12))

δ_O A threshold on the minimum overlap values (Equation (3.12))

$\mathcal{W}_E$ Proposed Effectiveness filter (Equation (3.13))

δ_E The required minimum positive change in M_E. (Equation (3.13))

$\mathcal{W}_S$ Proposed Stopwords filter (Equation (3.14))

S_{stop} A set of stopwords (Equation (3.14))

$\mathcal{F}_d(\phi_{q_i}, k')$ Function returning a list of the k' documents closest to embedding ϕ_{q_i}

Φ Set of feedback embeddings from f_b top-ranked feedback documents

υ_i A representative (centroid) embedding selected by applying KMeans among Φ

K Number of representative embeddings to select, i.e., number of clusters for KMeans

$\mathcal{F}_t(\upsilon_i, r)$ Function returning the r token ids corresponding to the r closest document embeddings to embedding υ_i

σ_i Importance score of v_i

F_e Set of expansion embeddings

f_e Number of expansion embeddings selected from K representative embeddings

f_b Number of feedback documents

β Parameter weighting the contribution of the expansion embeddings

F_e Set of expansion embeddings

Chapter 1

Introduction

One of the most fundamental information retrieval tasks is the *adhoc retrieval* task, where the search system provides a list of documents from a collection in response to the user's *information need* (Manning, 2009). Generally, to communicate with the search system, users express their information needs as questions or queries. Based on this user-initiated question or query, the search system returns a list of documents in descending order with respect to their estimated relevance to the query. However, an information retrieval system can fail when the relevant documents are hard to be retrieved by a user's query (Croft et al., 2010). One of the prominent causes behind a retrieval failure is the *under-specification* of the user's query, where the search query is too broad or ambiguous, and which makes it difficult for the search engine to accurately interpret the user's intent (Clarke et al., 2009). For instance, a search engine may fail to interpret the user's information need with a query 'coffee' as the "best coffee brands" or the "nearest coffee shop". This might happen because the user may not have enough knowledge or expertise in the topic they are searching for, thus resulting in ambiguous queries. Furthermore, even with well-formed user queries, a retrieval failure can also occur when there exists a *mismatch* between the language of the relevant documents and that of the user's query, which can be attributed to either lexical (or vocabulary) or semantic mismatches, or both. The lexical mismatch issue occurs when the words used in a query and a document conveying the same meaning do not match. This issue can be caused by various types of mismatches, such as synonym mismatch (e.g., 'computer' vs. 'laptop'), spelling mismatch (e.g., 'color' vs. 'colour'), stemming mismatch (e.g., 'look' vs. 'looking'), abbreviation mismatch (e.g., 'UK' vs. 'United Kingdom'), and others. On the other hand, a polysemous mismatch, where the word can express different meanings, is the root cause of the semantic mismatch problem. For example, the word 'bank' expresses different meanings in the phrases 'bank account' and 'river bank'.

For decades, sparse retrieval has been the dominant information retrieval architecture , where the query and documents are represented as sparse bag-of-words vectors with dimensions equal to the vocabulary size of the corpus. In practice, an inverted index is built to record the terms occurring in each document for fast retrieval. For search algorithms that rely on lexical matching, such as BM25 (Robertson et al., 1995), this can result in the lexical mismatch problem (Furnas et al., 1987). Pseudo-relevance feedback (PRF) techniques have been shown to be an effective approach to alleviate the vocabulary discrepancies between the user query and the relevant documents, by modifying the user's original query, typically by expanding and reweighing their relative importance with additional useful words. Many approaches follow this pseudo-relevance feedback paradigm – such as Rocchio's algorithm (Rocchio, 1971), the RM3 relevance language model (Abdul-Jaleel et al., 2004), or the DFR query expansion models (Amati and Van Rijsbergen, 2002) – where terms appearing in the top-ranked documents for the user's initial query are used to expand it. In common, these techniques assume the top returned documents from the initial retrieved document list are relevant, otherwise, there is a risk that the expanded terms drift the intent of the query. Alternatively, methods have been proposed for rephrasing user queries to generate paraphrases, aimed at addressing the issue of 'lexical mismatch'. These include the generation of lexical query substitutions, which identify phrases similar to a query based on resources like WordNet (Zukerman and Raskutti, 2002) or query logs (Jones et al., 2006).

In recent years, advanced pre-trained neural network models, such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020), have had a significant impact on information retrieval due to their ability to capture the latent traits of texts. Various effective neural rerankers, such as CEDR (MacAvaney et al., 2019), which is a BERT-based neural reranking model, have been proposed to promote highly relevant documents to the top position of the ranking. Pseudo-relevance feedback approaches have also demonstrated their efficacy when integrated with effective BERT-based neural reranking models, by providing a high-quality set of candidate documents obtained using the expanded query, which can then be reranked (Lin et al., 2021a). A few neural pseudo-relevance feedback techniques, such as CEQE (Naseri et al., 2021) and BERT-QR (Zheng et al., 2020) model, can further improve the retrieval effectiveness of sparse retrieval systems by alleviating the *lexical mismatch* problem using the expanded query (Abdul-Jaleel et al., 2004, Amati and Van Rijsbergen, 2002, Rocchio, 1971). However, as they rely on precise lexical matching over the inverted index, the *semantic mismatch* problem can still persist.

The recently proposed *dense retrieval* models alleviate the above semantic mismatch problem by encoding the query and document into contextualised embeddings, and have yielded significant improvements over lexical retrieval (Karpukhin et al., 2020, Khattab and Zaharia, 2020, Lin et al., 2020a, Xiong et al., 2021, Zhan et al., 2021). Indeed, by representing the query and document using the contextualised embeddings, relevant documents are typically retrieved based on the semantic matching between the query and document representations. In particular, two families of dense retrieval models have emerged (Macdonald et al., 2021b): the *single representation* dense retrieval models, where each passage and query are embedded using a single contextualised embedding, e.g. using BERT [CLS] token; and the *multiple representation* dense retrieval models, in which each token of the query or document are individually embedded using contextualised embeddings. In the single representation dense retrieval models, the relevance of a document to a query is estimated according to the inner product of their corresponding contextualised embeddings in the same vector space. On the other hand, multiple representation dense retrieval models employ a *late interaction* scoring mechanism to estimate a similarity score between the query and document. Compared to traditional sparse retrieval models, dense retrieval models are able to capture complex relationships between queries and documents, even when they have not been trained on domain-specific data. Thus, both in-domain (where the retrieval model is trained and evaluated on the data follow the same distribution) and zero-shot out-of-domain (where the retrieval model is trained and evaluated on data with different distribution) evaluations are prevalent for dense retrieval models.

To this end, these techniques, both the aforementioned sparse and dense retrieval models, only address some of the potential causes for the low retrieval effectiveness: lexical mismatch and semantic mismatch. However, the issue that the users' queries might be *underrepresented* still exists for both sparse retrieval and even with advanced dense retrieval systems (Wang et al., 2022c). Indeed, in the dense retrieval paradigm, even encoding the queries into the dense representation, with insufficient information provided, the query representation can still be underrepresented, for e.g. the ambiguous short user query: "Jaguar", which can refer to the luxury car brand Jaguar or the animal jaguar. Research about implementing pseudo-relevance feedback techniques, which are effective to address the underrepresented query problem (Xu and Croft, 2017), to perform query reformulation entirely operating in the dense retrieval paradigm is still in its infancy. Thus, this thesis examines the potential of the pseudo-relevance feedback information in the era of neural information retrieval, for both sparse and dense retrieval paradigms. Moreover, we incorporate external knowledge from high-quality external resources with dense pseudo-relevance feedback techniques and conduct semantic search on the dense index. Furthermore, we explore dense pseudo-relevance feedback techniques in both unsupervised and supervised manners.

1.1 Thesis Statement

The statement of this thesis is that pseudo-relevance feedback information can be used in neural-based models to improve retrieval effectiveness, for both sparse and dense retrieval; In particular, pseudo-relevance feedback information can be used by a sequence-to-sequence neural model to generate more effective query reformulations for sparse retrieval; Moreover, applying pseudo-relevance feedback on contextualised embeddings can refine the query representation for multiple representation dense retrieval, in particular, the ColBERT model; Furthermore, performing external pseudo-relevance feedback to refine the query representation can improve the zero-shot performance for both sparse and dense retrieval; In addition, our key ColBERT-PRF model can be effectively extended to various forms of dense-PRF models; Finally, pseudo-relevance feedback information can be used to train a deep language model-based feedback weighting model for identifying the discriminating expansion embeddings for query reformulation.

1.2 Thesis Contributions

To summarise, our contributions can be summarised as follows:

1. We alleviate the query and document vocabulary mismatch problem by employing the neural sequence-to-sequence neural models with the pseudo-relevance feedback information to perform query reformulation for sparse retrieval.

Classical pseudo-relevance feedback approaches, such as query expansion, select the expansion terms from the pseudo-relevant set directly. In Chapter 3, we show that the pseudo-relevance feedback set can act as context information of the query for a pre-trained language model, thus producing a refined query reformulation.

2. We propose a pseudo-relevant feedback mechanism in the multiple representation dense retrieval paradigm and propose ColBERT-PRF, which performs query expansion entirely over the dense index.

One challenge for the traditional pseudo-relevance feedback approaches as well as the neural-PRF techniques is that they rely on the inverted index, thus making it hard to address the *semantic mismatch* problem. In Chapter 4, we propose a dense pseudo-relevance feedback model named ColBERT-PRF. In particular, by implementing a pseudo-relevance feedback technique for dense retrieval and performing query expansion entirely in the embedding space, ColBERT-PRF effectively reduces both the lexical and semantic mismatch problem. Moreover, the refined query representation produced by ColBERT-PRF helps to clarify the user's information needs.

3. We incorporate external knowledge from high-quality external resources with dense pseudo-relevance feedback techniques and conduct semantic search on the dense index.

One challenge with dense retrieval models is their effectiveness on out-of-domain datasets in a zero-shot manner. Performing external pseudo-relevance feedback techniques can augment the zero-shot retrieval performance. However, the effectiveness of the dense pseudo-relevance feedback models, for both the single representation-based or multiple representation-based dense PRF models, on an out-of-domain dataset that has not been used to train it is still unclear. In addition, external expansion techniques on sparse retrieval have shown their effectiveness by bringing high-quality pseudo-relevance feedback from the external resource but have not been studied in the dense retrieval paradigm. Therefore, to improve the zero-shot performance of dense retrieval models, in Chapter 5, we incorporate external knowledge from high-quality external resources with dense pseudo-relevance feedback techniques and conduct the semantic search on the dense index.

4. We generalise ColBERT-PRF to Col$\star$-PRF model, which generalises ColBERT-PRF technique across various underlying pre-trained language models.

The underlying first and second pass of retrieval model only apply to BERT and WordPiece tokeniser, the effect of other various pre-trained language models and tokenisers are under-investigated in the literature. Therefore, in Chapter 6, we generalise our proposed ColBERT-PRF model to Col⋆-PRF models, which are built upon different pretrained language models. In addition, we thoroughly examined the semantic matching behaviour occurring within various Col⋆-PRF models.

5. We train a supervised feedback weighting model that takes the query and text of the pseudo-relevance feedback documents as input and outputs feedback weights for selecting and weighting the expansion embeddings.

Our proposed ColBERT-PRF model operates in an unsupervised way to perform dense query expansion and it relies on a heuristic method, e.g. statistics of the frequency of the tokens, to identify expansion embeddings. In Chapter 7, we provide a different method that performs dense query expansion in a supervised manner. In particular, we propose a supervised model that learns feedback weights, where for each pseudo-relevance feedback token, we construct a contrastive objective. With the contrastive objective, the feedback weighting model is trained to assign high weights to the tokens that are better to discriminate the relevant documents from the non-relevant documents. Therefore, the feedback weighting model identifies the discriminating expansion embeddings for semantic search

1.3 Origins of the Materials

Most of the material presented in this thesis is based on papers published in journals and conferences throughout this PhD programme:

- In (Wang et al., 2023c), we investigate two representative query reformulation frameworks, namely the GenQR and GenPRF frameworks. In particular, for each framework, we investigate different techniques, including fine-tuning and direct prompting, to harness the knowledge of the sequence-to-sequence language models. This work was published in ACM SIGIR2023 Gen-IR Workshop and contributes to Chapter 3.

- In (Wang et al., 2021a), we propose a dense pseudo-relevance feedback technique for multiple representation dense retrieval, called ColBERT-PRF. This work is proposed and its retrieval effectiveness is evaluated for the passage retrieval task in (Wang et al., 2021a), which was published at the ACM ICTIR 2021 conference. This work contributes to our Chapter 4.

- In addition, in the work of (Wang et al., 2022c), we further evaluate the retrieval effectiveness of ColBERT-PRF for long documents. Moreover, several effective ColBERT-PRF variants with different expansion embedding weighting methods and efficient ColBERT-PRF variants with the application of different clustering methods and/or the approximate nearest scoring method are investigated in this work. This work was published in the ACM TWEB journal and also contributes to Chapter 4.

- In (Wang et al., 2022b), we thoroughly investigate the retrieval effectiveness of an external pseudo-relevance feedback approaches for both sparse and dense retrieval in (Wang et al., 2022b). In particular, we examine the retrieval effectiveness for both the single and multiple representation pseudo-relevance feedback techniques with high-quality external resources. This work (Wang et al., 2022b) was published in the Elsevier IPM journal.

- In (Wang et al., 2023d), we extend the ColBERT to Col$\star$, by applying the contextualised late interaction mechanism upon different types of pre-trained language models and tokenisation techniques. In addition, it quantifies the semantic matching behaviour for the proposed Col$\star$ models. Moreover, we further quantify the semantic matching behaviour for both Col$\star$ and Col$\star$-PRF models. This work was published in ACM SIGIR 2023. In this work, we build the work of Chapter 6 upon (Wang et al., 2023d). More specifically, we further investigate the effectiveness of implementing pseudo-relevance feedback techniques on Col$\star$ models, i.e. Col$\star$-PRF models and measures the semantic matching behaviour for Col$\star$-PRF models in Chapter 6.

- In (Wang et al., 2023b), we proposed a deep language model-based feedback weighting model, called CWPRF (Wang et al., 2023b), which performs dense query expansion in a supervised way. In particular, CWPRF model is trained to learn to assign feedback weights to facilitate the expansion embeddings selection process for performing dense query expansion. In particular, the weighting model is trained to learn from the ability of the feedback embeddings to discriminate between relevant and non-relevant documents for semantic search. This work was published at the ACL 2023 conference and contributes to our Chapter 7.

1.4 Thesis Outline

The remainder of this thesis is organised as follows:

- Chapter 2 provides background knowledge for retrieval systems, from the sparse retrieval to the dense retrieval paradigms, as well as their related literature. In addition, we detail the implementations of pseudo-relevance feedback techniques for different retrieval systems, namely classical, neural-based, and dense pseudo-relevance feedback techniques.

- Chapter 3 introduces our proposed sequence-to-sequence query reformulation method. In particular, we investigate two representative query reformulation frameworks, GenQR and

GenPRF frameworks and for each proposed framework, we investigate both fine-tuning and direct prompting to explore the ability of different pretrained language models for query reformulation.

- Chapter 4 details our proposed ColBERT-PRF model, which conducts effective query reformulation operating entirely on the dense index. In particular, ColBERT-PRF implements the pseudo-relevance feedback mechanism within the late interaction dense retrieval paradigm. More specifically, based on the pseudo-relevant set of documents identified using a first-pass dense retrieval, ColBERT-PRF extract representative feedback embeddings, while ensuring that these embeddings discriminate among passages, which are then appended to the query representation.

- Chapter 5 explores the effectiveness of external pseudo-relevance feedback techniques for zero-shot retrieval tasks. On the one hand, we investigate the benefit of the classical sparse PRF model that is supported by a pseudo-relevant feedback set obtained from different dense retrieval approaches. Moreover, we study the effectiveness of external dense expansion on dense retrieval. This facilitates the investigation of how external expansion changes the semantic matching behaviour of retrieval. Finally, we investigate whether the sparse external retrieval can produce feedback information that is useful for dense retrieval.

- Chapter 6 investigates the generalisation of the contextualised late interaction mechanism upon various pre-trained models, such as RoBERTa (Liu et al., 2020) and ALBERT (Lan et al., 2020) and different tokenisation techniques, such as WordPiece (Schuster and Nakajima, 2012), BPE (Bostrom and Durrett, 2020) and SentencePiece (Kudo and Richardson, 2018). Furthermore, this chapter inspects the nature of matches occurring within the generalised contextualised late interaction mechanism. In particular, the matching behaviour on different token families. Finally, we quantify the contribution of lexical and semantic matching on retrieval effectiveness, respectively.

- Chapter 7 presents a novel technique, called CWPRF. Going further than our proposed unsupervised ColBERT-PRF model, CWPRF performs dense quey expansion in a supervised way. In particular, the CWPRF model is trained to to assign high expansion weights for tokens that can discriminate the relevant documents from the non-relevant documents. Based on the predicted weights, CWPRF helps to identify useful expansion embeddings for generating refined query representations.

- Chapter 8 provides concluding remarks and summarises the contributions of this thesis. Moreover, it discusses the limitations as well as presenting several future directions that build upon the foundation laid in this thesis.

Chapter 2

Background and Related Work

As discussed in Chapter 1, this thesis focuses on the adhoc retrieval task (Manning, 2009). A schematic view of the standard adhoc search task, as well as its components, is presented in Figure 2.1. In the adhoc retrieval task, the core component is the score function $s(q,d)$, which takes the query q and a document d as inputs and produces a score for this document with respect to its estimated relevance to the query. Based on the estimated relevance score for each document in the collection, we can obtain a ranked list with a descending relevance order for the query. Depending on the way the query and document are represented, the retrieval systems can be classified into two prevalent types: the *sparse* retrieval paradigm, where the query and document are represented into sparse vectors with a dimension size equal to the vocabulary size of the collection, and the *dense* retrieval paradigm, where the query and document are encoded into a low dimensional dense representation, such as that obtained from the BERT [CLS] embedding (Devlin et al., 2019).

In particular, in this chapter, we start with a description of the traditional sparse retrieval paradigm in Section 2.1, then Section 2.2 introduces the pretrained language models and their prevalent applications in information retrieval (IR), namely neural reranking models and document expansion, while the more advanced dense retrieval paradigm is detailed in Section 2.3. Next, we introduce pseudo-relevance feedback (PRF) techniques, specifically traditional, neural-based and dense PRF approaches in Section 2.4. Finally, retrieval evaluation methods and evaluation measures used in this thesis are detailed in Section 2.5.

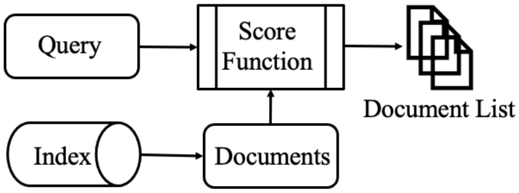


Figure 2.1: A schematic view of the adhoc search task.

2.1 Sparse Retrieval

In sparse retrieval systems, queries and documents are encoded into sparse high-dimensional representations. For instance, assume there are three documents in a collection:

Doc 1: A dog playing in the backyard.

Doc 2: A cute cat sleeps in the sun all day.

Doc 3: The brown fox is a cunning animal.

For these example documents, we can create a vocabulary, which consists of all the unique terms from all the documents and with a dimension as v. Based on this vocabulary, a binary vector (with a dimension size equal to v) for each document and query in the collection can be created. In particular, a query q and a document d can be represented by a vector of ordered index terms: $V_q = (\vec{q}_1, \vec{q}_2, ..., \vec{q}_{|v|})$ and $V_d = (\vec{d}_1, \vec{d}_2, ..., \vec{d}_{|v|})$, respectively. For instance, the sparse representation for Doc 2 in the above example is $V_{Doc2} = (1,1,0,0,0,1,0,1,0,0,1,0,0,1,1,0)$, where the length of the sparse vector equals to the size of the vocabulary, the value, 1 (or 0), indicates whether the term appears (or not) in the ordered vocabulary, e.g. in alphabetical order. similarly, for a query 'cute cat', the sparse query representation is $V_q = (0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$. These sparse representations, consisting of a bag of binary occurrence indicators, are often referred to as *one-hot encodings*. Next, we introduce the various prevalent sparse retrieval models in Section 2.1.1, starting with the vector-space and TF-IDF models, then the probabilistic model, exemplified by the BM25 model, and finally, the information-theoretic model, exemplified by the DPH model. Moreover, we describe how to experiment with the introduced retrieval models in practice and how to construct the more complex retrieval pipelines using the PyTerrier platform in Section 2.1.2.

2.1.1 Sparse Retrieval Models

Vector-space Retrieval Models The most straightforward way to estimate the relevance between a query representation V_q and a document representation V_d is by using similarity functions, such as the cosine similarity method, as follows:

$$s(q,d) = \text{cosine}(V_q, V_d) = \frac{\sum_{j=1}^{\text{vocab}} \vec{d}_j \cdot \vec{q}_j}{\sum_{j=1}^{\text{vocab}} ||\vec{d}_j|| \cdot ||\vec{q}_j||} \tag{2.1}$$

These models, based on the geometric representations of the query and document, are known as *vector-space* retrieval models (Salton et al., 1975). While vector-space models are easy to implement, they suffer from the high-dimensional representations and ignore term-dependence relationships. Indeed, in the web search scale, the dimension of a document representation can be thousands or even millions of dimensions. Low-quality tokens , such as the tokens correspond to the 'stopwords' or low IDF tokens, are often removed to reduce the vocabulary size (Scholer et al., 2002, Tonellotto et al., 2013).

Furthermore, the vector-space model can be adapted to various weighting schemes, for instance, the term frequency *tf* and the inverse document frequency *IDF* are often used to describe the importance of a term. More generally, the relevance score of a query-document pair can be estimated using the following score function:

$$s(q,d) = \sum_{t \in q \cap d} f(\eta(q,t), \eta(d,t)), \tag{2.2}$$

where $f(t)$ represents a score function calculated based on the overlapping terms between document d and query q. In particular, the query and document can be represented as $\eta(q,t)$ and $\eta(d,t)$ respectively using their associated statistics, such as term frequency *tf* and document frequency *df*.

TF-IDF Retrieval Model: Now we introduce the simplest retrieval model instantiated from Equation (2.2), i.e. the TF-IDF ranking model (Sparck Jones, 1972). TF-IDF combines the count of index term occurrences in a document, i.e. tf and the inverse document frequency, i.e. idf. In particular, tf assigns a higher weight to terms that appear more frequently while idf measures the rareness of a term across a collection of documents and gives high weights for terms that occur rarely in the collection. There are many variations of idf, however a simple variant can be described as follows:

$$idf(t,d) = \log\left(\frac{N+1}{N_i+1}\right), \tag{2.3}$$

where N_i is the number of documents containing the token t_i and N is the total number of documents in the collection; +1 is applied for smoothing to avoid division by zero. Based on this, the weighting scheme for TF-IDF can be described as follows,

$$s_{\textbf{TF-IDF}}(q,d) = \sum_{t \in q \cap d} f(\eta(q,t), \eta(d,t)) = \sum_{t \in q \cap d} tf(t,d) \cdot idf(t,d), \tag{2.4}$$

where $tf(t,d)$ refers to the frequency of the term t in the document d and $idf(t,d)$ is the inverse document frequency.

Okapi BM25 Retrieval Model: Besides the vector space weighting model, which makes the explicit usage of the sparse representations, probabilistic retrieval models, which are based on the probability theory, have been extensively used in the literature (Robertson, 1977, Robertson and Spärck Jones, 1976). For instance, BM25 (Robertson et al., 1995, Spärck Jones et al., 2000) was proposed to incorporate the query and document term weights into the scoring function and is still dominant and one of the best-performing sparse ranking models today. Therefore, in this thesis, we extensively use BM25 in the experimental part as a strong sparse retrieval baseline ranking model and as a robust basis for creating more advanced neural baseline models. BM25 is defined as Equation 2.8, where N_i is the number of documents containing the token t_i. $tf(t,d)$ and $tf(t,q)$ denote the frequency of the term t in the document and query, respectively, k_2, k_1 and b are free parameters, where k_1 and k_2 control how much the term frequency is scaled and b controls the normalisation of the document length. *dl* and *avgdl* denote the length of a document

d and the average length of the documents in the collection, respectively.

$$\begin{aligned}
s_{\mathrm{BM25}}(q,d) &= \sum_{t \in q \cap d} f(\eta(q,t), \eta(d,t)) && (2.5)\\
&= \sum_{t \in q \cap d} idf(t,d) \cdot \eta(q,t) \cdot \eta(d,t) && (2.6)\\
&= \sum_{t \in q \cap d} idf(t,d) \cdot \frac{tf(t,q)(1+k_2)}{k_2 + tf(t,q)} \cdot \frac{tf(t,d)(k_1+1)}{tf(t,d) + k_1(1-b+b\frac{dl}{avgdl})} && (2.7)\\
&= \sum_{t \in q \cap d} \log\left(\frac{N - N_i + 0.5}{N_i + 0.5}\right) \cdot \frac{tf(t,q)(1+k_2)}{k_2 + tf(t,q)} \cdot \frac{tf(t,d)(k_1+1)}{tf(t,d) + k_1(1-b+b\frac{dl}{avgdl})}. && (2.8)
\end{aligned}$$

DPH Retrieval Model: As we can see from the previous section, BM25 contains several free parameters (k_1, k_2, b) that can require further tuning depending on the retrieval task. Here we introduce a parameter-free ranking model called DPH (Amati et al., 2007). DPH is developed under the information-theoretic-based Divergence From Randomness (DFR) framework (Amati, 2006), which assumes that: *The more the divergence of a term's within-document term-frequency from its frequency within the whole collection, the more the information carried by the word t in the document d (Amati, 2006, Amati et al., 2007).* In particular, the parameter-free DPH weighting model is calculated as follows,

$$s_{\mathrm{DPH}}(q,d) = \sum_{t \in q \cap d} \eta(q,t) \cdot \eta(d,t), \tag{2.9}$$

where $\eta(q,t) = \frac{tf(t,q)}{max_{t_i \in q} tf(t_i,q)}$, and $\eta(d,t) = \frac{tf(t,d)(1-\frac{tf(t,d)}{l_d})^2}{tf(t,d)+1} \log(tf(t,d) \cdot \frac{\ln}{l_d \cdot (tf(t,C)}) + 0.5 \cdot \log(2\pi \cdot tf(t,d) \cdot (1 - \frac{tf(t,d)}{l_d}))$. As the parameter-free DPH provides an effective and efficient alternative of BM25 (Robertson et al., 1995, Spärck Jones et al., 2000), in this thesis, we also employ DPH as one of the baseline sparse ranking models.

2.1.2 Retrieval Pipelines

In Section 2.1.1, we introduced several classical sparse retrieval models. In recent years, with the advent of neural network models, the application of machine-learning-based applications in the information retrieval field has experienced a great growth. Researchers tend to perform more complex end-to-end experiments, such as learning to rank (LTR) and neural reranking in a retrieval pipeline. Thus, in this thesis, we resort to expressive high-level notations for describing complex retrieval pipelines. For instance, we use the + operator to establish a linear combination of the scores of the two retrieved results lists, a feature supported by the PyTerrier platform (Macdonald and Tonellotto, 2020, Macdonald et al., 2021a). We describe the advanced complex pipelines using PyTerrier, for neural reranking in Section 2.2, for dense retrieval in

Table 2.1: PyTerrier transformers (Macdonald et al., 2021a).

Input		Output	Transformer
Q	→	Q′	Query rewriting
Q	→	R	Retrieval
R	→	Q′	Query expansion
R	→	R′	Re-ranking
R	→	R_f	Feature extraction

Table 2.2: PyTerrier operators for combining transformers (Macdonald et al., 2021a).

Operator	Name	Description
»	*then*	Pass the output from one transformer to the next transformer
+	*linear combine*	Sum the query-document scores of the two retrieved result lists
*	*scalar product*	Multiply the query-document scores of a retrieved result list by a scalar
**	*feature union*	Combine two retrieved result lists as features
|	*set union*	Make the set union of documents from the two retrieved result lists
&	*set intersection*	Make the set intersection of the two retrieved result lists
%	*rank cutoff*	Shorten a retrieved result list to the first *k* elements
ˆ	*concatenate*	Add the retrieved result list from one transformer to the bottom of the other

Section 2.3 and for integrating pseudo-relevance feedback in Section 2.4.

PyTerrier introduces transformers and operators to perform complex retrieval tasks by constructing retrieval pipelines, enabling experiments to be conducted in a declarative manner. In particular, Table 2.1 presents the classes of PyTerrier's transformers, which are distinguished by their input and output. Table 2.2 lists the operators for combining the transformers to build retrieval pipeline. More formally, let $Ret(I,k)(q) \rightarrow R$ denotes a retrieval process that takes a query q as input and returns a ranked list of k documents as the retrieved set R by searching over the index I.[1] In addition, based on the introduced PyTerrier transformers, let $\overset{x}{»}$ denote passing the output of one transformation process, with type x, as input to another, where $x = R$ indicates the type is a ranking of documents and $x = Q$ indicates the type is a query. As we will show, this definition is sufficiently general enough to encompass classical sparse retrieval models (e.g. BM25 or DPH) and dense retrieval models (in Section 2.3) as well as pseudo-relevance feedback settings (in Section 2.4). Based on this, the simplest retrieval pipeline for retrieval using BM25 can be created as $\text{Ret}_{\text{BM25}}(I,k)$. Moreover, to handle very long documents, a more complex retrieval pipeline can first break down each retrieved document into passages. Then a scorer can be applied to the smaller text chunks. Finally, the document score can be aggregated over the text chunk scores using a MaxPassage function. Such a retrieval pipeline can be created as follows,

$$\texttt{Pipe}_{\texttt{MaxP}} = \text{Ret}_{\text{Retriever}}(I,k') \overset{R}{»} \text{Sliding}(I, \texttt{length}, \texttt{stride}) \overset{R}{»} \text{Ret}_{\text{Scorer}}(I,k) \overset{R}{»} \text{MaxPassage}(\cdot)\ , \tag{2.10}$$

[1] To aid readability, we use 'document' and 'passage' interchangeably in our explanations of the models.

where Sliding(·) denotes the sliding window function that is typically applied on long documents to be split into shorter texts. More specifically, two parameters are used in the sliding window function: length and stride, where the length determines the number of words in each chunk while the stride controls the number of words the window moves over the text chunk for each step. Typically, if the stride is equal to the length, the window would create non-overlapping chunks. In addition, the MaxPassage(·) function is applied to aggregate the scores of the chunked passages for each document. This pipeline is particularly useful when applying the advanced BERT-based reranker and we will discuss this design in Section 2.2.

Overall, in this section, we introduced the classical sparse representation method and various prevalent traditional term weighting models in Section 2.1.1. Then, we described the experimental platform we used, namely PyTerrier, throughout this thesis in Section 2.1.2. However, all the retrieval models introduced in Section 2.1 rely on the sparse representations of the query and documents. The sparse bag-of-words representations have drawbacks, such as high dimensionality, the term independence assumption and ignoring the order of the words as well as the context of the words. Distributed semantic models (Le and Mikolov, 2014, Mikolov et al., 2013), such as the Word2Vec model (Le and Mikolov, 2014), encode each vocabulary entry to a low-dimensional word representation. Word2Vec embeddings can capture the syntactic and semantic word relationships between words in the representation space. For example, in the Word2Vec representation space, ‘apple’, ‘banana’, and ‘orange’ are all fruits in the Word2Vec representation space, so they should be placed close to each other but far away from the word representations of ‘queen’ and ‘king’. Although Word2Vec embeddings have demonstrated promising results for natural language understanding (Baroni et al., 2014, Le and Mikolov, 2014), Word2Vec model is unable to capture the dynamic contextualised meaning of a word. Indeed, since the Word2Vec model produces one vector for each word, even for different sense words with the same surface word form, such as ‘bank’ in ‘bank of river’ vs. ‘bank of Scotland’. Later, pretrained language models, such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020), have led to the development of natural language understanding and semantic search techniques. More importantly, models such as BERT can produce contextual word embeddings that take the surrounding context into consideration, thus producing different embeddings of the ‘bank’ in different contexts. In the following section, we introduce various families of pretrained language models and their usage in modern information retrieval systems.

2.2 Pretrained Language Models (PLMs)

In this section, we first introduce various types of pretrained language models depending on their underlying transformer architecture in Section 2.2.1. Then we detail the application of PLMs in the retrieve-then-rerank pipeline and the usage of PLMs for refining the document representations in Section 2.2.2 and Section 2.2.3, respectively.

2.2.1 Transformer-based PLMs

For decades, deep convolutional and recurrent neural networks (CNNs and RNNs) have been widely used (Cho et al., 2014, Hochreiter and Schmidhuber, 1997, LeCun et al., 1998). However, these models have difficulties learning contextualised word representations. More recently, Vaswani et al. (2017) proposed a novel self-attention deep-learning model, called Transformers. The transformer architecture consists of large encoder and decoder blocks to process the data. In particular, the encoder maps an input sequence to a continuous representation for each input element. The continuous representations are then used by the decoder to generate an output sequence. Based on this, various advanced pretrained models are developed based on the Encoder-only or Decoder-only transformer or both. Most of these pretrained models can be obtained from the Huggingface [2] library, which is established as the state-of-the-art library for transformer-based PLMs.

Encoder-based PLMs: Encoder-based PLMs consist only of stacks of transformer encoder layers, thus the output for the encoder-based PLM models can be regarded as the contextualised representations of the input sequence. Based on the depth of the model architecture, i.e. the number of transformer layers consisting in the encoder, different sizes of the PLM models can be trained. One of the salient encoder-based model families is BERT (Devlin et al., 2019), which has various types of models that differ with their model size, from $\text{BERT}_{\text{tiny}}$ to $\text{BERT}_{\text{Large}}$. BERT employs masked language modelling (MLM) pretraining techniques, where some input tokens are replaced with `[MASK]` tokens and the model is trained to reconstruct the masked tokens. In addition, popular encoder-based PLMs including XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2020), ELECTRA (Clark et al., 2020) or ALBERT (Lan et al., 2020). Encoder-based PLMs have also been applied for various IR tasks, where various cross-encoder reranker models have been developed based on the encoder-based PLMs, as well as dense retrieval models that benefit from the contextualised representations produced by the encoder-based PLMs. Among various encoder-based PLMs, BERT is the most used PLM in IR, hence, we illustrate the input and output of BERT in Figure 2.2. In particular, for an input sequence, the special tokens `[CLS]` and `[SEP]` are prepended and appended, respectively. Then BERT produces a dense embedding for each input token. In Section 2.2.2, we further describe the cross-encoder reranker models and the dense retrieval paradigms in Section 2.3.

Decoder-based PLMs: Different from encoder-based PLMs, decoder-based PLMs only employ stacks of the multi-head attention and feed-forward transformer decoder layers. In general, the decoder-based PLMs consist of a series of GPT models, including from GPT-1 (Radford et al., 2018), GPT-2 (Radford et al., 2019) or GPT-3 (Brown et al., 2020).

Encoder-Decoder based PLMs: Encoder-decoder based PLMs maintain most of the whole transformer architecture but are more powerful after pre-training on a diverse set of tasks and data. A number of Encoder-Decoder PLMs have been developed, such as the T5 (Raffel et al.,

[2] `https://huggingface.co/`

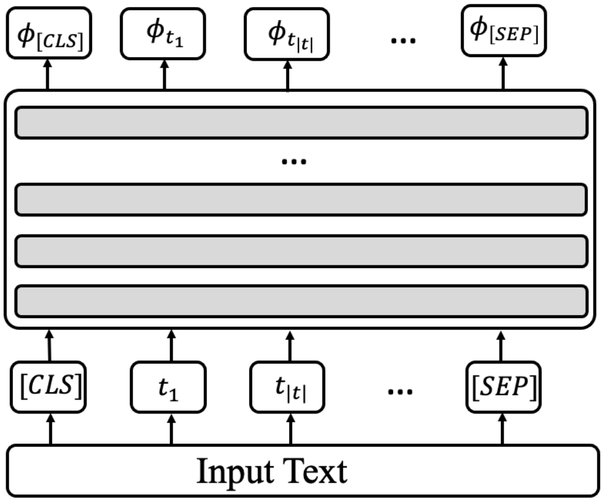


Figure 2.2: Illustration of the input and output of BERT.

2020) or BART (Lewis et al., 2020) models. The T5 model has also been used for document expansion in IR, which we will further detail in Section 2.2.3. In this thesis, we also demonstrate that T5 can also be used to perform adhoc query expansion tasks.

Tokenisation of PLMs Tokenisation is an important technique to preprocess the input text before input to a contextualised language model. In particular, as transformer-based models learn representations for each unique token, a limited-size vocabulary is important. Indeed, a large vocabulary size would cause increased memory and time complexity (Jean et al., 2015), and difficulty in learning accurate representations for rare tokens (Mikolov et al., 2013). For these reasons, sub-word tokenisation is usually used to split the input text into small chunks of text. Thus, frequently used words are given unique IDs, while rare words will be processed into sub-words. Prevalent tokenisation techniques used by large pretrained language models include the WordPiece (Schuster and Nakajima, 2012), Byte-Pair Encoding (BPE) (Sennrich et al., 2016) and SentencePiece (Kudo and Richardson, 2018) tokenisation techniques. For instance, WordPiece (Schuster and Nakajima, 2012) is used by BERT (Devlin et al., 2019) and miniLM (Wang et al., 2020a); BPE (Sennrich et al., 2016) is used by the RoBERTa (Liu et al., 2020) and GPT (Brown et al., 2020, Radford et al., 2018, 2019) models; SentencePiece (Kudo and Richardson, 2018) is used by the ALBERT (Lan et al., 2020) and T5 (Raffel et al., 2020) models. In particular, the BPE (Sennrich et al., 2016) and WordPiece (Schuster and Nakajima, 2012) tokenisation technique merge frequently occurring character sequences into larger tokens but control the vocabulary size using different algorithms to maximise the likelihood of the training data. In contrast, SentencePiece initially treats each whole sentence as one large token and learns to split it into sub-words.

In practice, *Transfer Learning* technique is often used to harness the knowledge stored in the pretrained models. Transfer learning, where the knowledge stored for addressing one task can be transferred to address other related tasks, provides flexible ways to leverage the knowledge encapsulated in large pretrained models, such as T5. As a result, many researchers have turned to make use of such knowledge to address various downstream tasks, such as using the T5 model for document expansion, which will be detailed in Section 2.2.3, and using a BERT model for neural query expansion, which will be introduced in Section 2.4.2.

We introduced the different types of PLMs depending on their transformer architecture, i.e. Encoder-only, Decoder-only and Encoder-Decoder PLMs. In addition, we described various types of tokenisation techniques used by PLMs. In this thesis, we are more concerned with the applications of these PLMs in the IR field, namely the adhoc search task. In particular, Section 2.2.2 introduces how to employ PLMs for the text ranking task, where the PLMs are regarded as the classifier. Moreover, Section 2.2.3 introduces the scenarios where PLMs are employed as the document expansion models.

2.2.2 Multi-Stage Retrieval using PLMs

The advanced pretrained neural network models introduced in Section 2.2, such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020), have shown to have the ability to capture the latent traits of texts. Recently, the PLMs have been successfully applied in various IR tasks, such as fine-tuning the PLMs for text classification, as the ranking model or as a useful representation model. One of the most successful applications of PLMs is as an effective reranker, where the model reorders a list of documents in accordance with their estimated relevance to the search query. However, while PLMs have proven to be highly effective as rerankers, they typically require significant amounts of time and GPU resources to implement due to their complex architecture and large size, making it impractical to apply them to the entire index. Thus, some efficient methods, such as traditional sparse retrieval models BM25 or DPH (cf. Section 2.1), are typically employed to produce a set of candidates retrieved from the index. Then, a more powerful reranking model is applied to further improve the effectiveness of the ranking of the candidate documents. In general, this retrieval pipeline is called *Multi-Stage Ranking* or *Retrieve-then-Rerank* retrieval, which can be expressed as follows:

$$\texttt{Pipe}_{\texttt{Multi-Stage}} = \text{Ret}_{\texttt{Retriver}}(I,k') \overset{R}{\gg} \text{Ret}_{\texttt{Scorer/Reranker}}(I,k). \tag{2.11}$$

In the following, we introduce two types of popular reranking models based on the Encoder-only transformer and the Encoder-Decoder transformer PLMs.

Encoder-only PLMs for Reranking: A Encoder-based PLMs can take a pair of query and document as the input and produce the relevance score of the document for the query. The Encoder-only text reranking models are also referred to as *Cross-Encoder* rerankers. One of the earliest neural rerankers is monoBERT proposed by Nogueira and Cho (2019), where the BERT model is fine-tuned for passage reranking. Specifically, the input query q and document d are first split and truncated to a list of word pieces $q = q_1, q_2, \cdots, q_{|q|}$ and concatenated together

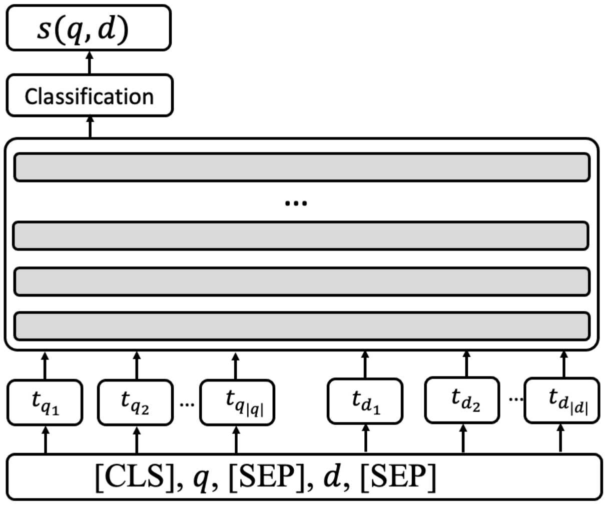


Figure 2.3: Cross Encoder Retrieval Model.

as the input of the MonoBERT model. More formally, the input for the BERT model can be expressed as BERT(`[CLS]`, $q_1, ..., q_{|q|}$, `[SEP]`, $d_1, ..., d_{|d|}$, `[SEP]`). Then, the `[CLS]` vector of the final output layer of BERT is obtained and input to a single-layer classification network to produce the estimated relevance score indicating the input document's relevance to the query. The inference process for the Encoder-only PLMs reranker can be depicted in Figure 2.3. As the input of the Encoder-only reranking model takes the document appended to the query, the self-attention mechanism within the transformer layers is applied to every query and document token. Note that the input text length for training BERT is typically set to 512. Therefore, for the document retrieval task, where the document length is longer than 512, more tricks are needed. One straightforward technique is to divide the long document into smaller chunks and the maximum score is aggregated as the final score of the document. This model is called MaxP (Dai and Callan, 2019a). On the other hand, MacAvaney et al. (2019) proposed a BERT-based reranking model called CEDR for document reranking, which incorporates BERT into the neural document ranking model. Moreover, Gao et al. (2021b) explored a bag of tricks to improve the training of the BERT-based reranking model via hard negative samples mining and a Localized Contrastive Estimation (LCE) loss. Later, Pradeep et al. (2022) replaced BERT with the ELECTRA PLM within this training framework thus obtaining a new more effective reranker named monoELECTRA. In sum, the difference between these Encoder-only PLMs reranking models is that each model is built upon different pretrained language models and employs different training techniques to train. Overall, these cross-encoder rerankers work based on the contextualised dense embeddings of the documents produced by the (Encoder-only) pretrained language model. Instead, in the following, we detail various reranker models that are working on the text generation models based on the Encoder-Decoder structure.

Encoder-Decoder PLMs as Rerankers Besides the Encoder-only reranking models, the Encoder-Decoder type of PLMs have also been explored as text rerankers. They can also fit into

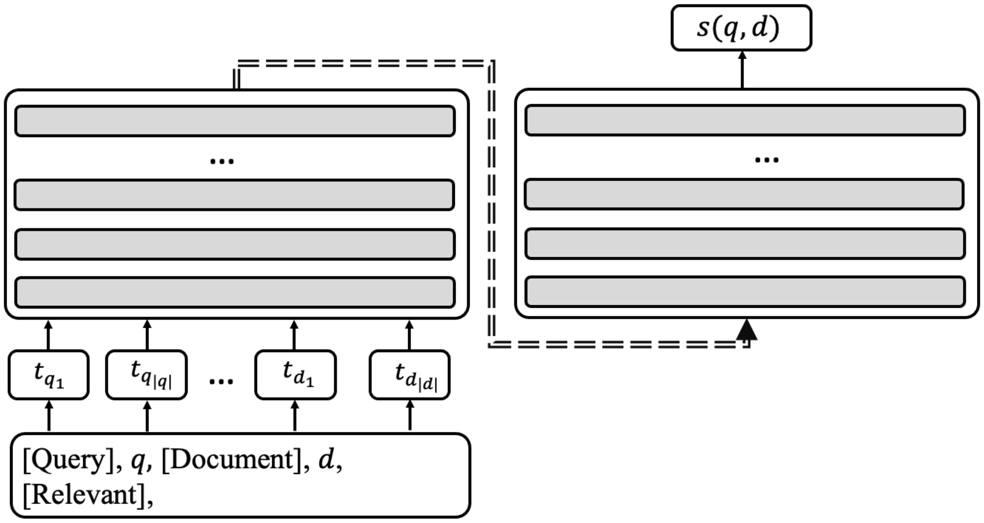


Figure 2.4: The Sequence-to-Sequence Retrieval Model.

the multi-stage retrieval pipeline shown in Equation (2.11). Encoder-Decoder PLMs as Rerankers include models such as monoT5 (Nogueira et al., 2020) and RankT5 (Zhuang et al., 2022). More specifically, monoT5 is built upon the T5's Sequence-to-Sequence framework and is trained to generate the relevance labels, such as 'True' or 'False' as target tokens, for the input query and document pair. The input template for monoT5 is:

$$\text{T5}(\texttt{[Query]}, q_1, ..., q_{|q|}, \texttt{[Document]}, d_1, ..., d_{|d|}, \texttt{[Relevant]}), \tag{2.12}$$

where the `[Query]` and `[Document]` and `[Relevant]` are special tokens indicating the beginning of the query and document tokens as well as the predicted relevance target token, such as 'True' or 'False', respectively. Then, the estimated relevance score is obtained by applying a softmax function on the logits underlying the target tokens. Overall, the architecture for the Sequence-to-Sequence retrieval models can be illustrated in Figure 2.4.

2.2.3 Refined Document Representations Obtained using PLMs

Besides using PLMs for text ranking, another strand of research leverages PLMs to perform *document expansion* trying to alleviate the pervasive vocabulary mismatch problem in information retrieval. Document expansion augments each document with supplementary information, for instance, additional terms selected from a corpus or predicted queries generated by a natural language generation model from the original document. DeepCT (Dai and Callan, 2020b) and HDCT (Dai and Callan, 2020a) learn each term importance score of a term using the BERT model in a context-aware manner for both passages and documents. In contrast, the Doc2query approach expands each document by generating multiple predicted queries using a Sequence-to-Sequence transformer model (Nogueira et al., 2019b) and later applied the T5 model (Nogueira et al., 2019a) for the same purpose. This is effective but comes with the significant upfront cost of applying the expensive PLM, for instance, the T5 model, to every document before indexing. Later, DeepImpact (Mallia et al., 2021) further estimates the semantic impact score of the token of a document enriched by the DocT5Query model, which can be regarded as a combination of the DeepCT and DocT5query methods.

All the above methods focus on enriching documents on the term level, which means these PLMs-based models are still based on bag-of-words representations, which are indexed using sparse vectors, cf. Section 2.1. In contrast, a thread of works focused on augmenting the document representations. EPIC (MacAvaney et al., 2020b) directly predicts the contextualised importance and creates expanded document representations. In particular, EPIC employs a BERT model to project each term in the passage as a dense vector with a dimension as the whole BERT WordPiece vocabulary size, and then the document representation is aggregated using the maximum score for each term. The final passage representation is in the whole vocabulary size, which has the effect of augmenting the original passage representation. In addition, SparTerm (Bai et al., 2020) also employs the BERT model to directly learn the weights for the sparse representation. It first learns a BERT-based encoder model to produce a dense importance distribution with the dimension of the whole vocabulary size and then employs a gating controller to control the term activation to ensure that the final representation contains the original passage terms. Later, SPLADE (Formal et al., 2021a), as well as SPLADE-v2 (Formal et al., 2022), augment the performance of SparTerm by applying a log-based saturation function to smooth the over-dominated terms and training with additional in-batch negative samples. More recently, COIL (Gao et al., 2021a) learns contextualised exact matching vectors for each token, which can be encoded into an inverted index. uniCOIL (Lin and Ma, 2021) simplifies COIL to learn a scalar weight instead of the contextualised matching vector for each document token.

These models all enrich the document's original representation by augmenting the document representation in a whole vocabulary size. Therefore, these models are often referred to as *learned sparse representation* models. In general, the enriched document representations of these document expansion and learned sparse models are compatible with the traditional inverted index structure. As a result, they are often employed as first-stage effective sparse retrievers, like BM25, within a multi-stage retrieval pipeline using a cross-encoder for reranking the final results.

To summarise, this section began with the introduction of the Transformer architecture and three types of pretrained language models classified by their underlying transformer architecture. Then, we discussed the application of the PLMs for text ranking (cf. Section 2.2.2) and for refining the document representation (cf. Section 2.2.3) in IR. We note that all these models rely on the inverted index in different formats. In the following, in Section 2.3, we will further discuss the application of the PLMs for performing adhoc ranking entirely on a dense index. Moreover, besides refining document representation, in Section 2.4, we will introduce various pseudo-relevance feedback techniques for refining query representation.

2.3 Dense Retrieval

The above retrieval models, namely the sparse retrieval (Section 2.1), multi-stage retrieval (Section 2.2.2) and document expansion models as well as the learned sparse retrieval models (Section 2.2.3) all rely on the traditional sparse inverted index to different extents. Indeed, sparse retrieval relies on the vocabulary-sized sparse representation-based inverted index; multi-stage retrieval requires a high recall and also an efficient first round of retrieval, which is typically deployed on the more efficient sparse inverted index, to provide a candidate set for reranking. Moreover, although learned sparse models incorporate the contextualised importance of each term into the index, the format of the indices is still a sparse representation-based inverted index. This means that although efficient for storage and computation, these sparse representation methods may lose some semantic relationships between words.

Based on these considerations, another strand of works have directly delved into retrieval models based on the dense document representations. Unlike sparse representations, *dense representations*, such as those obtained by using BERT embeddings, can capture the semantic relationships between words and encode information in a lower dimensional continuous vector sparse. In addition, dense representations allow for an efficient similarity computation between the query and the document's dense representations. In terms of the model structure, different from the popular "cross-encoder" based BERT-rerankers (MacAvaney et al., 2019, Nogueira and Cho, 2019), dense retrieval models usually apply a BERT-based *bi-encoder* (or *dual-encoder*) structure. The query and document are encoded separately into dense representations. Then, dense retrieval performs relevance scoring through the encoded contextualised representations of queries and documents. Moreover, according to the way the queries and the documents are encoded, dense retrieval models can be divided into two families (Macdonald et al., 2021b): single representation and multiple representation dense retrieval models. Section 2.3.1 and Section 2.3.2 describe the dense retrieval paradigm and the salient works for single and multiple representation dense retrieval, respectively.

2.3.1 Single Representation Models

In single representation models, each query or document as a whole is encoded into a single dense representation, i.e. a single embedding for each query or passage. Then the relevance between the query and document is estimated based on the encoded query and document vectors, for example using their dot product similarity. There have been many dense retrieval models developed since the beginning of 2020, such as DPR (Karpukhin et al., 2020), RepBERT (Zhan et al., 2020), ANCE (Xiong et al., 2021), Rocket-QA (Qu et al., 2021) and TCT-ColBERT (Lin et al., 2020a). In general, these models share a similar model structure and inference process but are differentiated by their training process.

More formally, in terms of the model structure, the single representation dense retrieval models consist of a BERT-based query encoder E_Q and a BERT-based document encoder E_D.

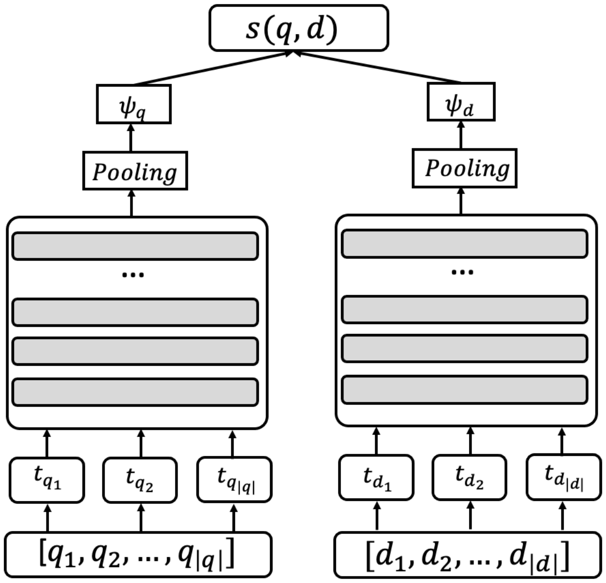


Figure 2.5: Architecture of the Single Representation Dense Retrieval.

The input template for obtaining the query and document representations using E_Q and E_D can be described as follows:

$$\psi_q = \mathrm{E_Q}(\texttt{[CLS]}, q_1, ..., q_{|q|}, \texttt{[SEP]}) \in \mathbb{R}^{1\times m}, \tag{2.13}$$

and

$$\psi_d = \mathrm{E_D}(\texttt{[CLS]}, d_1, ..., d_{|d|}, \texttt{[SEP]}) \in \mathbb{R}^{1\times m}. \tag{2.14}$$

Then, the similarity score between the query and document is estimated as follows:

$$s(q,d) = sim(\psi_q, \psi_d), \tag{2.15}$$

where $sim(.,.)$ denotes the similarity function used to measure the similarity between the query and document embeddings. This is commonly instantiated with the L2-based or cosine similarity functions. Figure 2.5 depicts the architecture of the single representation dense retrieval paradigm. The pooling layer can be instantiated by different functions depending on the specific model design, such as the `[CLS]` pooling, Max or Average pooling functions. We further describe this in the following specific model introduction.

With a trained model, during the indexing time of a single representation dense model, the document representations are pre-computed and encoded in the dense index. The retrieval pipeline of single representation dense retrieval model deployed on the PyTerrier platform is illustrated in Figure 2.6 and can be expressed as follows:

$$\texttt{Pipe}_{\texttt{Single-Rep}} = \mathrm{E_Q} \overset{R}{\gg} \texttt{Ret}_{\mathrm{ANN}}(I, k), \tag{2.16}$$

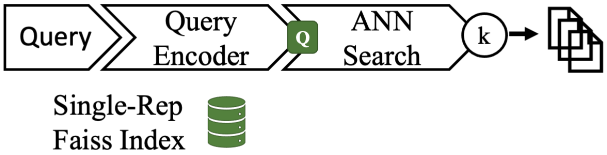


Figure 2.6: Conceptual Architecture of Single Representation Dense Retrieval using PyTerrier.

where $\texttt{E}_{\texttt{Q}}$ denotes the query encoder of the dense retrieval model, which does not rely on the index and encodes the query text into the dense query embedding using the trained dense retrieval model. $\texttt{Ret}_{\texttt{ANN}}(I,k)$ search denotes performing an Approximate Nearest Neighbour search method on its dense index and returns k documents. Furthermore, in the following, we describe several salient retrieval models, namely the DPR and ANCE models.

DPR: Dense Passage Retriever (DPR) was proposed by Karpukhin et al. (2020) for open-domain question-answering tasks. The query encoder and the passage encoder are employed as BERT (Devlin et al., 2019) networks and BERT's `[CLS]` representations are taken as the query and passage representations. Accordingly, the pooling layer in Figure 2.5 can be instantiated to the `[CLS]` pooling function in DPR. Thus, the similarity between the query and passage representations is defined as follows:

$$s(q,p) = \mathrm{sim}(\psi_{q_{[CLS]}}, \psi_{p_{[CLS]}}) = \psi_{q_{[CLS]}}{}^{\top} \cdot \psi_{p_{[CLS]}}, \tag{2.17}$$

where *sim* denotes the similarity function for measuring the relevance between the query and passage representations, which commonly employ the dot-product or cosine similarity functions. DPR uses the dot-product similarity function as its similarity function.

During training, each training instance contains one query q, one positive (relevant) passage p^+, and DPR mines n negative (non-relevant) passages $p^-_{1\cdots n}$. Then DPR is trained using the negative log-likelihood loss, which is used to make the similarity score higher for positive documents and lower for negative documents, as follows:

$$\mathcal{L}(q,p^+,p_{1\cdots n}{}^-) = -\log \frac{\exp[\mathrm{sim}(\psi_{q_{[CLS]}}, \psi^+_{p_{[CLS]}})]}{\exp[\mathrm{sim}(\psi_{q_{[CLS]}}, \psi^+_{p_{[CLS]}})] + \exp[\sum_{j\in n} \mathrm{sim}(\psi_{q_{[CLS]}}, \psi_{p^-_{j_{[CLS]}}})]}. \tag{2.18}$$

In particular, Karpukhin et al. (2020) experimented with negative (non-relevant) passages that were selected from three different sources: (1) random, where the negative samples are selected randomly from the training corpora; (2) BM25, where the non-relevant negative samples in the BM25 retrieved list are selected as the negative samples; and (3) in-batch negatives, where the passages from the other instances in the same training batch are regarded as the negative samples. Empirically, the in-batch negative sampling method has been shown to be the most effective among the above sampling techniques (Lin et al., 2021a).

ANCE: Approximate Nearest Neighbour Negative Contrastive Estimation (ANCE) was proposed by Xiong et al. (2021) and also falls into the single representation dense retrieval

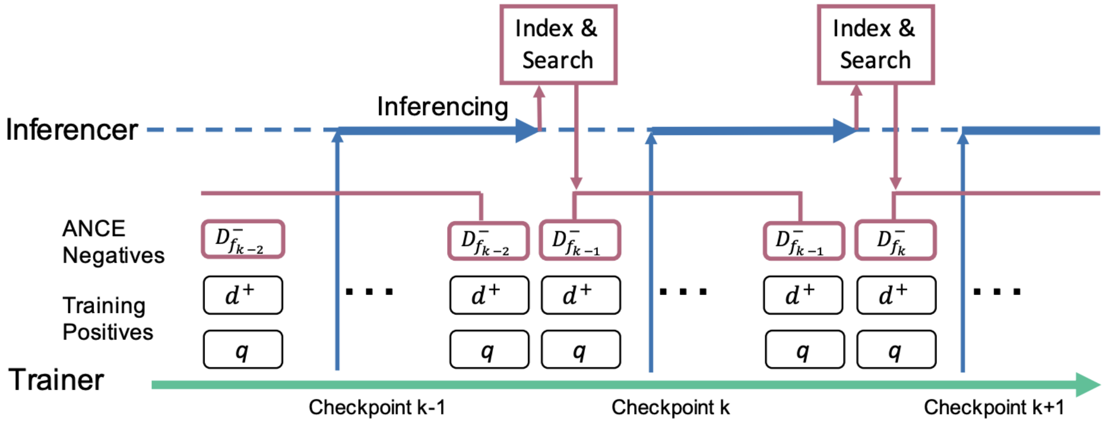


Figure 2.7: ANCE Asynchronous Training (Xiong et al., 2021).

paradigm depicted in Figure 2.6. similar to DPR, the ANCE model adopts the bi-encoder architecture, and the query documents are encoded as BERT's `CLS` embeddings. The relevance score between the query and passage pair can be estimated using Equation (2.17). In contrast to DPR, ANCE theoretically analyses that negative samples which can be easily distinguished from the positive samples, such as the hard negative samples identified from the BM25 retrieved list, can lead to diminishing gradient norms. Therefore, Xiong et al. (2021) argued that mining hard negative samples via ANN search running on an ANCE index can effectively train the ANCE dense retrieval model. In particular, ANCE employs an asynchronous training method, where it repeatedly mines hard negative samples from the ANN search returned list and updates the ANCE dense index with the most newly trained ANN model. Figure 2.7 presents this asynchronous training procedure of ANCE.

As ANCE is one of the more effective than DPR as a single representation dense retrieval model (Lin et al., 2021a), in this thesis, we not only employ ANCE as a baseline of our proposed models but also build on top of ANCE using our own proposed PRF techniques to further improve the retrieval effectiveness on single-representation dense retrieval models in Chapter 5. We use $\texttt{Ret}_{\texttt{ANCE}}(I,k)$ to denote the ANCE retrieval model.

2.3.2 Multiple Representation Models

In contrast to single representation, multiple representation dense retrieval models, exemplified by ColBERT (Khattab and Zaharia, 2020, Santhanam et al., 2022), encode each token of the query or document into a dense representation. ColBERT consists of a query encoder E_Q and a document encoder E_D, which are fined-tuned based on the pretrained BERT model.

The queries and documents are represented by tokens from a vocabulary V. The input query tokens are encoded as a list of query embeddings (each of dimension m) using E_Q, as follows:

$$\phi_q = \mathrm{ColBERT}(\texttt{[CLS]}, \texttt{[Q]}, q_1, ..., q_{|q|}) \in \mathbb{R}^{32\times m}, \tag{2.19}$$

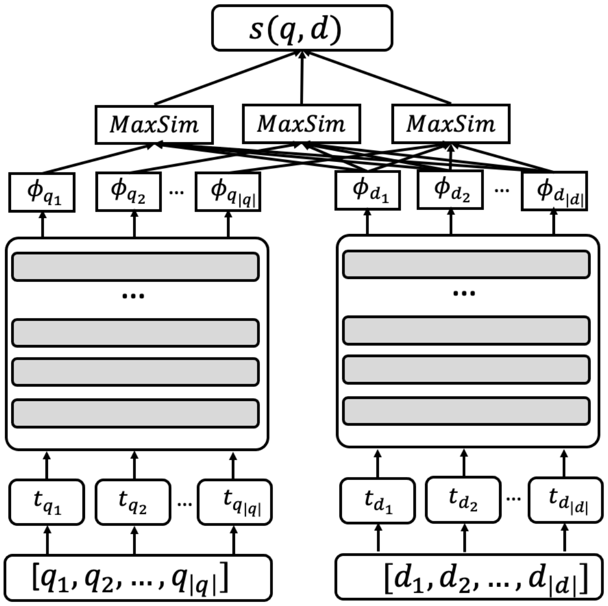


Figure 2.8: Contextualised Late Interaction in Multiple Representation Dense Retrieval.

where $m = 128$ and '[MASK]' embeddings are used to pad the input query embeddings to 32. Similarly, for a document d, we encode it into a list document embeddings using E_D, as follows:

$$\phi_d = \text{ColBERT}(\texttt{[CLS]},\texttt{[D]},d_1,...,d_{|d|}) \in \mathbb{R}^{|d|\times m}. \tag{2.20}$$

The number of encoded query tokens is fixed to $|q| = 32$ and filled with the special token '`[MASK]`' if the original query contains less than 32 tokens. Moreover, a linear layer is used to map the BERT representations into a low-dimensional vector with m components, typically $m = 128$ (Khattab and Zaharia, 2020).

To estimate the relevance score of a document to a query, ColBERT implements a two-stage scoring pipeline: in the first stage, an approximate nearest neighbour (ANN) search produces a set of candidate passages. The ANN search used here is similar to the scoring process of the single representation dense retrieval models depicted in Figure 2.5. In the second stage, these passages are re-ranked with a so-called late interaction mechanism. In particular, this late interaction scoring mechanism is illustrated in Figure 2.8.

More formally, the relevance score of a document d to a query q, denoted as $s(q,d)$, is calculated using a late interaction matching mechanism. The late interaction mechanism is based on the bag of encoded query and document representations, where the maximum similarity score among all the document representations for each query token representation is calculated and then summed to obtain the final relevance score:

$$s_{\text{Maxsim}}(q,d) = \sum_{i=1}^{|q|} \max_{j=1,...,|d|} sim(\phi_{q_i}^T, \phi_{d_j}), \tag{2.21}$$

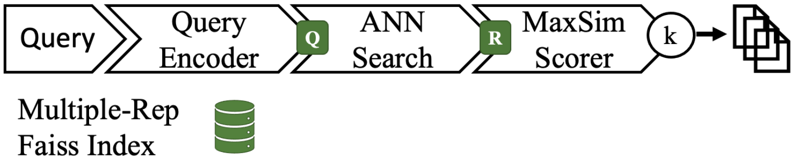


Figure 2.9: Conceptual Architecture of Multiple Representation Dense Retrieval using PyTerrier.

similar to single representation dense retrieval, there are several commonly used similarity functions $sim(.,.)$ for dense retrieval models, namely the L2-based and Cosine similarity functions (Khattab and Zaharia, 2020, Santhanam et al., 2022). When experimenting on the PyTerrier platform, the retrieval pipeline for ColBERT can be expressed using Equation (2.22) and depicted in Figure 2.9.

$$\texttt{Pipe}_{\texttt{Multi-Rep}} = \mathrm{E}_{\mathrm{Q}} \overset{R}{\gg} \texttt{Ret}_{\texttt{ANN}}(I,k') \overset{R}{\gg} \texttt{Maxsim}(I,k), \tag{2.22}$$

where $\texttt{Ret}_{\texttt{ANN}}(I,k')$ denotes performing an Approximate Nearest Neighbour search method on the multiple representation dense index and returns k' documents. $\texttt{Maxsim}(I,k)$ denotes performing the Maximum similarity scoring method on its dense index and returns k documents.

To summarise, this section introduced both single and multiple dense retrieval models. For single representation, we detailed the model structure, training and inference process for the DPR, ANCE and TCT-ColBERT models. For multiple representations, we highlight the working procedure for the ColBERT model. Besides different types of retrieval paradigms, namely single and multiple representation models, these models also distinguish themselves by using various training techniques, such as negative sampling, hard negative sampling and knowledge distillation etc.

To the best of our knowledge, ColBERT (Khattab and Zaharia, 2020) exemplifies the implementation of an end-to-end IR system that uses multiple representations of query and document. The ColBERT model is a scalable yet expressive neural retrieval model, and has been used in various ways to enhance retrieval effectiveness. For instance, TCT-ColBERT (Lin et al., 2020a, 2021b, Wang et al., 2022a) employs ColBERT as the teacher model and distils the learned knowledge from ColBERT to a single representation dense retrieval model. In addition, several works focus on improving the retrieval efficiency of ColBERT. For instance, pruning techniques on query (Tonellotto and Macdonald, 2021a) and document embeddings (Acquavia et al., 2023) that are estimated as less useful are found to achieve more efficient retrievals. Moreover, XTR (Lee et al., 2023) simplifies the scoring component of ColBERT which only uses the retrieved passage tokens rather than all tokens.

Furthermore, upon comparing the single and multiple representation dense retrieval paradigms, we observe that ColBERT outperforms other single representation dense retrieval models like ANCE and TCT-ColBERT in several key aspects. Firstly, the multiple representation dense retrieval model, ColBERT, often exhibits higher retrieval effectiveness than the single representation dense retrieval models, such as ANCE (cf. Section 2.3.1). Secondly, it is easier to access and further extract useful information from the retrieved document embeddings in the multiple representation dense retrieval paradigm than in the single representation dense retrieval paradigm. Finally, ColBERT enables us to inspect the matching behaviour occurring within the dense retrieval. ColBERT employs the above late interaction scoring mechanism (computed using Equation (2.21)) to estimate a similarity score between the query and document. In practice, some embeddings will represent the same token – a *lexical match* – while others may match at a *semantic* level ('car' vs. 'vehicle'). In particular, the extent that such *semantic matching behaviour* occurs among the contextualised representations is still under-investigated. Indeed, it is difficult to disentangle the semantic matching from the dot product operation on the single-representation dense retrieval models introduced in the previous sections. However, multiple-representation dense retrieval models provide the possibility to inspect the dense matching behaviour and to investigate the nature of its effectiveness improvements.

Based on these discussions, in this thesis, we focus on building our work upon the multiple representation dense retrieval paradigm. In particular, we investigate how to effectively implement the pseudo-relevance feedback mechanism on the multiple representation dense retrieval paradigm using the token-level embedding representations. In the following section, we introduce the various pseudo-relevance feedback technique families, namely the sparse PRF, neural PRF and Dense PRF approaches, which correspond to the aforementioned retrieval paradigms.

2.4 Pseudo-Relevance Feedback

As we discussed in Chapter 1, the classical sparse retrieval models introduced in Section 2.1 can fail when there is a lexical or semantic mismatch between the query and the relevant document(s), or when the user's query is under-specified. Recently, various advanced PLM-based rerankers introduced in Section 2.2 and the dense retrieval models introduced in Section 2.3 have demonstrated their ability to improve retrieval effectiveness by alleviating semantic mismatch, as the underlying PLMs are capable of capturing the semantic relationship between the query and document. In addition, several document expansion models introduced in Section 2.2.3 attempt to address the lexical mismatch problem by expanding the original document with potentially relevant information, such as questions that the document may answer.

However, we argue that among all the aforementioned retrieval paradigms, namely the sparse retrieval, retrieve and rerank, and dense retrieval paradigms, user queries can still be under-represented. Indeed, users may lack sufficient knowledge about the topic they are searching for, which can make it challenging for them to formulate well-represented queries using the appropriate keywords or phrases (Carmel and Yom-Tov, 2010). Many researchers seek to reformulate the user's initial query into an articulated query for more effective retrieval. In particular, query reformulation can generally be expressed as a process $\mathcal{P}$ that takes the user's initial query q^0 and converts it into a new representation q^r, with the aim of improving retrieval effectiveness:

$$q^r = \mathcal{P}(q^0, ...), \tag{2.23}$$

where "..." denotes the optional information that may be used by a query reformulation process $\mathcal{P}$. For instance, a query rewriting process might rely on the text of the query alone:

$$q^r = \mathcal{P}_{QR}(q^0). \tag{2.24}$$

On the other hand, many query expansion approaches obtain a reformulated query through the application of a classical pseudo-relevance feedback mechanism – such as Rocchio (Croft et al., 2010) or RM3 (Abdul-Jaleel et al., 2004) – which makes use of query terms occurring in the top-ranked documents returned for the original query q^0, as follows:

$$q^r = \mathcal{P}_{PRF}(q^0, R_k(q^0)), \tag{2.25}$$

where $R_k(q^0)$ is a ranking of k documents obtained using q^0 and q^r takes the form of a weighted set of terms.

The pseudo-relevance feedback (PRF) paradigm generally consists of three stages: (i) the initial retrieval stage, where an initial retriever is employed to provide a list of documents and the top-returned documents are assumed to be relevant to the input query; (ii) the PRF stage, which identifies the useful expansion terms from the pseudo-relevant documents and appended them to the initial query to produce a refined query representation; (iii) the reranking stage, where a reranking process is conducted based on the refined query representation to recalculate the similarity score of the document to the input query and reorder accordingly.

Following the definition of the PyTerrier operators introduced in Section 2.1.2, pseudo-relevance feedback techniques can be generally described as follows: given some set of ranked documents R, let a pseudo-relevance feedback process be denoted as $PRF(I, \theta)(R) \to q$, where θ are the parameters of the process. Typically, a PRF process takes the top-ranked documents from the initial retrieved document list as the pseudo-relevance feedback documents, extracts useful information from the PRF documents and in turn, produces a reformulated query. When implementing a typical PRF technique, a retrieval pipeline can be established as,

$$\texttt{Pipe}_{\texttt{PRF}} = \texttt{Retriever}(I, k') \overset{R}{\gg} \texttt{PRF}(I, \theta) \overset{Q}{\gg} \texttt{Reranker}(I, k), \tag{2.26}$$

In addition, Figure 2.10 depicts the retrieval pipeline of a PRF process using PyTerrier.

In the following, we further detail the working process of the traditional sparse PRF techniques working within the sparse retrieval paradigm in Section 2.4.1, including the Rocchio algorithm, the RM3 algorithm and the DFR Bo1 algorithm. Moreover, in Section 2.4.2, we introduce several salient neural-based PRF techniques which can be deployed within the retrieval-and-rerank

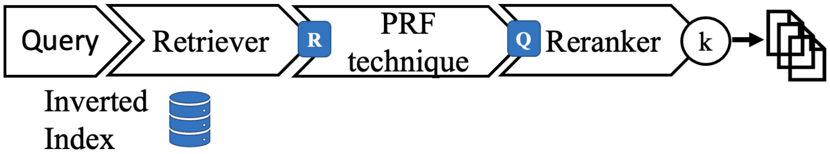


Figure 2.10: Conduct traditional sparse PRF experiment using PyTerrier.

retrieval paradigm. Finally, we focus on more advanced dense PRF techniques in this thesis and introduce several concurrently proposed dense PRF models in Section 2.4.3.

2.4.1 Sparse PRF Approaches

Many pseudo-relevance feedback techniques have been proposed to address the vocabulary mismatch problem in the sparse retrieval paradigm (cf. Section 2.1) and we refer these PRF approaches to sparse PRF approaches. Such sparse query expansion approaches, which rewrite the user's query, have been shown to be an effective approach to alleviate the vocabulary discrepancies between the user query and the relevant documents, by modifying the user's original query to improve the retrieval effectiveness. In the following, we introduce several salient sparse PRF methods, namely the Rocchio relevance feedback algorithm, RM3 and Bo1 algorithms.

Rocchio Algorithm: The Rocchio algorithm was proposed by Rocchio (1971). It operates in the vector space model and modifies the initial query vector to get closer to the average vector of the relevant documents while moving away from the average vector of the non-relevant documents. More specifically, the optimised query vector is obtained by averaging the document vectors of the relevant documents and adding to the original query vector and at the same time subtracting the mean averaged non-relevant document vectors from it. More formally, this process can be described as follows,

$$\vec{q^r} = \alpha \cdot \vec{q_0} + \frac{\beta}{\mid D_r \mid} \sum_{d_i \in D_r} \vec{d_i} - \frac{\gamma}{\mid D_n \mid} \sum_{d_i \in D_n} \vec{d_i}, \tag{2.27}$$

where D_r and D_n denote the known relevant and non-relevant sets of the documents and $\vec{q_0}$ and $\vec{q^r}$ denote the initial user query and the optimised new query, respectively. Tunable parameters: α weights the initial query, β weights relevant documents and γ weights the non-relevant documents. The Rocchio technique assumes that we can obtain a user-judged set of relevant and non-relevant documents in order to improve the query representation. However, in practical applications, users may be reluctant to make explicit relevant judgements for retrieved documents. Instead, the pseudo-relevance feedback formulation assumes that the top-ranked documents of an initial retrieved document list corresponding to a query are relevant, while the lower-ranked documents can be considered non-relevant. In the following, we further detail two salient pseudo-relevance feedback term weighting models, namely the RM3 and Bo1 algorithms.

RM3: Now, we introduce the RM3 (Abdul-Jaleel et al., 2004) model. For each candidate expansion term t, we first measure its association to a query $q = q_1, \ldots, q_{|q|}$ using the joint probability of observing the word together with the query words, i.e. $P\left(t, q_1, \ldots, q_{|q|}\right)$. The candidate word t can be sampled from the set of pseudo-relevance feedback documents denoted as PRF. Thus, the association score for each candidate expansion term t is calculated by

$$\begin{aligned} S_{RM3}(t) &= P\left(t, q_1, \ldots, q_{|q|}\right) \\ &= \frac{1}{\#PRF} \sum_{d \in PRF} \left(\frac{tf(t,d)}{|d|} \times \prod_{i=1}^{|q|} \frac{tf(q_i,d) + \mu P(q_i|C))}{|d| + \mu}\right), \end{aligned} \tag{2.28}$$

where $\mu = 2500$ is a smoothing parameter. Based on this, the expansion terms are selected according to the calculated association score and then added to the original query. Finally, the term weighting formula is expressed as follows,

$$\begin{aligned} Score(t) &= \alpha * Score_{exp}(t) + (1-\alpha) * Score_{orig}(t) \\ &= \alpha * \frac{S_{RM3}(t)}{\sum_{d \in PRF} \sum_{t' \in d} S_{RM3}(t')} + (1-\alpha) * \frac{tf(t,q)}{|q|}, \end{aligned} \tag{2.29}$$

where $S_{RM3}(t)$ is calculated using Equation (2.28) and $0 \leq \alpha \leq 1$ controls the contribution of the expansion terms with respect to the original query terms like the α and β parameters in Equation (2.4.1). Indeed, although effective, there is a risk of the pseudo-relevance feedback techniques that the added terms *drift* the intent of the query. To address this issue, RM3 introduces a parameter α to balance the importance of additional terms derived from the feedback to the original query terms, which helps it avoid the issue of "query drift". RM3 technique has been widely used in the IR field, therefore, we also employ RM3 as a baseline model to compare with in this work.

Bo1: Bo1 (Amati and Van Rijsbergen, 2002) model is a query expansion model based on Bose-Einstein statistics. Bo1 is an effective variant of the Divergence From Randomness (DFR) term weighting model but articulates the query representation by internally performing the query expansion. In particular, for each candidate expansion term t, a score is calculated by

$$S_{Bo1}(t) = \left(\sum_{d \in PRF} tf(t,d)\right) * \log_2\left(\frac{1 + tf_{avg}(t,C)}{tf_{avg}(t,C)}\right) + \log_2(1 + tf_{avg}(t,C)), \tag{2.30}$$

where PRF denotes the set of pseudo-relevance feedback documents and $tf_{avg}(t,C)$ denotes the average term frequency of the term t in the whole collection, which is calculated as $tf_{avg}(t,C) = \sum_{d \in C} tf(t,d)/N$ and N is the number of documents in the collection. Based on the calculated scores obtained using Equation (2.30), we select the expansion terms to add to the original query, to formulate the final expanded query as follows:

$$\begin{aligned} Score(t) &= Score_{exp}(t) + Score_{orig}(t) \\ &= \frac{1 + log(tf(t,q))}{1 + \max_{t' \in q} log(tf(t',q))} + \frac{S_{Bo1}(t)}{\max_{t' \in d \in PRF} S_{Bo1}(t')}. \end{aligned} \tag{2.31}$$

The retrieval pipeline for experimenting with a specific PRF technique, such as Bo1 algorithm or RM3, can be achieved by instantiating the `PRF` technique as shown in Equation (2.26) and Figure 2.10 using a specific query expansion technique.

To summarise, in this section, we introduced several salient query expansion models, from Rocchio's relevance feedback technique to the RM3 relevance model and the Bo1 query expansion technique. These techniques are based on different assumptions but all rely on the sparse representations of the query and documents. Due to their effective retrieval capabilities and ease of interpretation, these techniques continue to play a crucial role in information retrieval. Therefore, in this thesis, we also employ RM3 and Bo1 models as baseline models for sparse PRF.

2.4.2 Neural PRF Approaches

With the advent of neural networks and pretrained language models such as BERT, various effective PLM-based rerankers have benefited from the contextual capabilities of PLMs. As a result, several PRF techniques have been proposed that attempt to take the contextualised meaning of the terms into consideration within a PRF process. Next, we detail two recently proposed neural PRF techniques, namely CEQE (Naseri et al., 2021) and BERT-QE (Zheng et al., 2020).

CEQE: Traditional PRF (pseudo relevance feedback) techniques, such as RM3 (Equation (2.28)) and Bo1 (Equation (2.30)), identify expansion terms based on statistical measures, such as term frequency. In contrast, a thread of works (Imani et al., 2019, Kuzi et al., 2016, Roy et al., 2018, 2016, Zamani and Croft, 2017) focuses on selecting the expansion terms based on their semantic similarity to the original query terms in a stable embedding space, such as Word2Vec. Later, with the advent of the BERT model (cf. Section 2.2.1), a contextualised embeddings-based query expansion model called CEQE (Naseri et al., 2021) was proposed. In particular, CEQE maps the feedback terms to their BERT embedding space and thus can identify the query-related terms that may not be captured by the statistical measures alone. Similar to RM3, CEQE is also conducted based on probabilistic language modelling. Its difference with RM3 is that: RM3 is based on the static lexical matching of the terms, but CEQE is based on the contextualised BERT representations. As introduced in Section 2.2.1, BERT uses the WordPiece tokeniser to split the input text into subwords. Therefore, a term can be split into several word pieces, for instance, the term "goldfish" is breakdown into "gold" and "##fish". Accordingly, CEQE employs three different methods, namely CEQE-Max, CEQE-Mul and CEQE-Centroid to

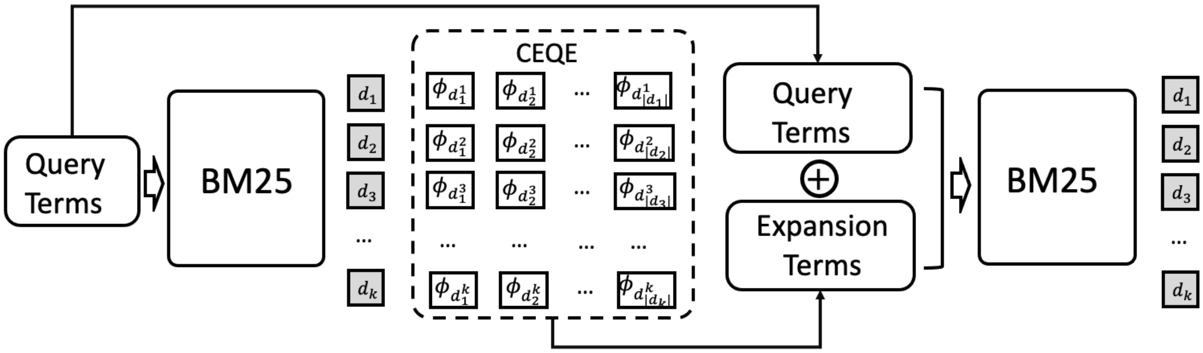

Query Terms
BM25
d_1
d_2
d_3
...
d_k
CEQE
$\phi_{d_1^1}$ $\phi_{d_2^1}$ ... $\phi_{d^1_{|d_1|}}$
$\phi_{d_1^2}$ $\phi_{d_2^2}$ ... $\phi_{d^2_{|d_2|}}$
$\phi_{d_1^3}$ $\phi_{d_2^3}$ ... $\phi_{d^3_{|d_3|}}$
$\phi_{d_1^k}$ $\phi_{d_2^k}$ ... $\phi_{d^k_{|d_k|}}$
Query Terms
⊕
Expansion Terms
BM25
d_1
d_2
d_3
...
d_k

Figure 2.11: Architecture of the CEQE approach.

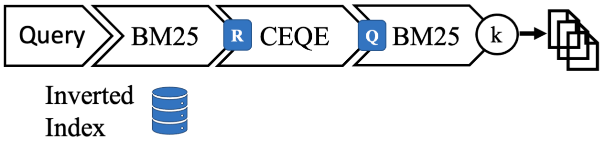


Figure 2.12: Conduct CEQE using PyTerrier.

transform the BERT token-level scores to the term-level scores. More specifically, CEQE-Max takes the max pooled subword relevance as the terms' similarity to the entire query, while CEQE-Mul multiplies subword relevance scores as the final relevance score of the term. In addition, CEQE-Centroid employs the centroid subword representations to measure the similarity of a term to the query. Then, the identified expansion terms are highly weighted and are added to the original query terms to form a new query.

Overall, the PRF process conducted by CEQE can be illustrated in Figure 2.11. From Figure 2.11, we can see that, although CEQE leverages the BERT contextualised representations during the expansion terms selection process, it still relies on a conventional sparse initial retrieval, i.e., BM25, and therefore the refined query representation needs mapping back in the lexical form. This means that the contextual meaning of an expansion term is lost - for instance, a polysemous word added to the query can result in a topic drift. The retrieval pipeline for conducting CEQE query expansion is illustrated as Figure 2.12 and can be constructed as follows:

$$\texttt{Pipe}_{\texttt{CEQE}} = \texttt{BM25}(I,k') \overset{R}{\gg} \texttt{CEQE}(I,\theta) \overset{Q}{\gg} \texttt{BM25}(I,k') \overset{R}{\gg} \texttt{Reranker}(I,k). \tag{2.32}$$

NPRF & BERT-QE: NPRF (Li et al., 2018) uses neural ranking models, such as DRMM (Guo et al., 2016) and KNRM (Xiong et al., 2017), to score the similarity of a document to a top-ranked feedback document. BERT-QE is conceptually similar to the NPRF model, but it measures the similarity of each document w.r.t. feedback chunks that are extracted from the top-ranked feedback documents. In particular, BERT-QE uses a cross-encoder to identify the feedback chunks that are similar to the original query and incorporate the passage relevance to the identified chunks into the relevance scoring function. More specifically, BERT-QE also consists of three stages: (i)

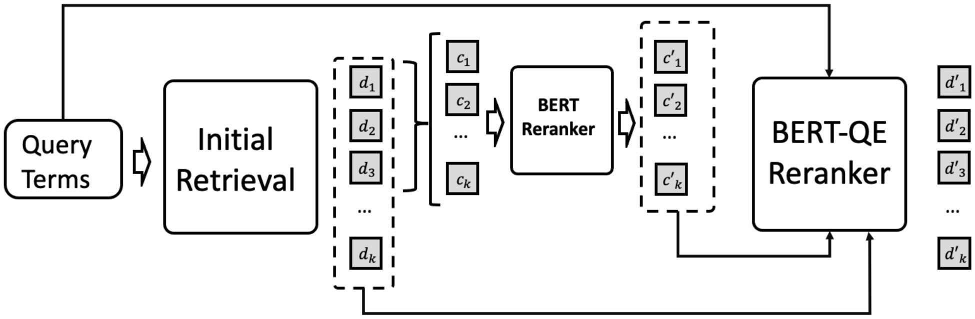


Figure 2.13: Architecture of the BERT-QE approach.

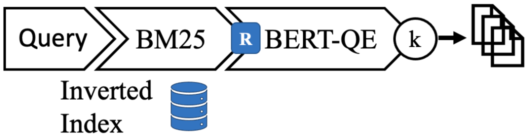


Figure 2.14: Conduct BERT-QE using PyTerrier

the initial retrieval stage; (ii) identifying the relevant information from PRF docs; and (iii) the BERT-QE reranking stage which leverages 3 sources of information: query, document and the selected PRF information in stage (ii).

The overall architecture of BERT-QE is illustrated in Figure 2.13. In particular, in stage (ii), BERT-QE selects the top 3 documents as the PRF documents and applies a sliding window to decompose the selected PRF docs into overlapping chunks. Then a pretrained BERT model is employed to score each of the chunks and the top-ranked chunks are regarded as the relevant feedback information. Then, in stage (iii), BERT-QE makes use of the selected relevant chunks together with the original query text to score a given document. When experimenting with BERT-QE on PyTerrier, the retrieval pipeline is constructed as follows and illustrated in Figure 2.14.

$$\texttt{Pipe}_{\texttt{BERT-QE}} = \texttt{BM25}(I,k') \overset{R}{\gg} \texttt{BERT-QE}(I,k). \tag{2.33}$$

The complex deployment of BERT-QE would result in an expensive application of many BERT computations – approximately 11× as many GPU operations than a BERT-based cross-encoder reranker (Zheng et al., 2020). Both Neural PRF and BERT-QE approaches leverage contextualised language models to rerank an initial ranking of documents retrieved by a preliminary sparse retrieval system. However, they cannot identify any new relevant documents from the collection that were not retrieved in the initial ranking.

To summarise, the aforementioned neural pseudo-relevance feedback techniques can effectively increase retrieval effectiveness when applied on a sparse retriever, like BM25 or DPH. Therefore, in this thesis, we employ both CEQE and BERT-QE as baseline models. On the one hand, most of the existing neural-PRF models are BERT-based approaches, the usefulness of the Sequence-to-Sequence models for query reformulation is still unclear. Therefore, in this thesis, we investigate the Sequence-to-Sequence based PLMs for effective query reformulation. On the other hand, despite effectiveness, the previously introduced neural-PRF models still operate on a sparse index that relies on precise lexical matching, specifically CEQE generates a sparse query while BERT-QE only functions as a reranker. As a consequence, the semantic mismatch caused by synonymous words remains unaddressed by these models. Inspired by dense retrieval (cf. Section 2.3), which can alleviate the semantic mismatch issue, and the effectiveness of pseudo-relevance feedback techniques, this thesis focuses on further refining the query representation for dense retrieval. Concurrently, other Dense-PRF efforts have also been attempted, and we detail these methods in the next section.

2.4.3 Dense PRF Approaches

Concurrently to our work, ANCE-PRF (Wang et al., 2021a, Yu et al., 2021b) was proposed, which can improve effectiveness within the single representation dense retrieval paradigm. Similar to ANCE-PRF, Vector-PRF (Li et al., 2021a) also refines the query representation heuristically, such as applying average pooling or a weighted combination of the query and the pseudo-relevance feedback embeddings. In the following, we detail the ANCE-PRF (Yu et al., 2021b) and Vector-PRF (Li et al., 2021a) techniques.

ANCE-PRF: ANCE-PRF (Li et al., 2021b, Yu et al., 2021b) has been proposed to perform PRF in a supervised manner, by re-encoding the query with the text of the PRF documents into a revised query vector. ANCE-PRF builds upon ANCE (Xiong et al., 2021), which we introduced in Section 2.3.1. In effect, ANCE-PRF is a cross-encoder that takes a query and all the pseudo-relevance feedback passages as input and outputs a single reformulated query embedding. More specifically, there are three stages in ANCE-PRF:

(i) An initial retrieval of ANCE is conducted, matching passages by the similarity of their single embedded representations with the single embedded representation of the query, which returns a list of k' pseudo-relevance feedback passages.

(ii) In the second PRF stage, ANCE-PRF fetches the top-k ranked documents and tokenises each PRF document into a list sequence of PRF tokens $p_1, p_2, ..., p_{|p|}$. Then the PRF tokens are appended to the original query tokens to form the new query and ANCE-PRF freezes the document encoder while training a query encoder alone. With a trained ANCE-PRF query encoder, ANCE-PRF produces a refined query representation. Specifically, the input template for the ANCE-PRF is formulated as follows, $\psi'_q = \mathrm{E}_{\mathrm{ANCE-PRF}}(\texttt{[CLS]}, q_1, \cdots, q_{|q|}, \texttt{[SEP]}\, p^1_1, \cdots, p^1_{|p^1|}, \texttt{[SEP]}, p^2_1, \cdots, p^2_{|p^2|}, \cdots \texttt{[SEP]}, p^k_1, \cdots, p^k_{|p^k|}, \texttt{[SEP]}) \in \mathbb{R}^{1\times m}$, where $\mathrm{E}_{\mathrm{ANCE-PRF}}$ denotes the ANCE-PRF query encoder and ψ'_q denotes the refined query representation by ANCE-PRF.

(iii) Then, in the third stage, another round of ANCE retrieval is performed based on the refined query representation. The similarity score between the refined query representation and the document is estimated as follows:

$$s_{\texttt{ANCE-PRF}}(q,d) = sim(\psi'_q, \psi_d), \tag{2.34}$$

where ψ_d can be obtained using the standard ANCE document encoder. This allows the refined query representation to be used with the original ANCE index without modification. When experimenting with ANCE-PRF on PyTerrier, the retrieval pipeline can be constructed as Equation (2.35) and illustrated in Figure 2.15.

$$\texttt{Pipe}_{\texttt{ANCE-PRF}} = \texttt{Ret}_{\texttt{ANCE}}(I,k') \overset{R}{\gg} \texttt{PRF}_{\texttt{ANCE}}(I,\theta) \overset{Q}{\gg} \texttt{Ret}_{\texttt{ANCE}}(I,k). \tag{2.35}$$

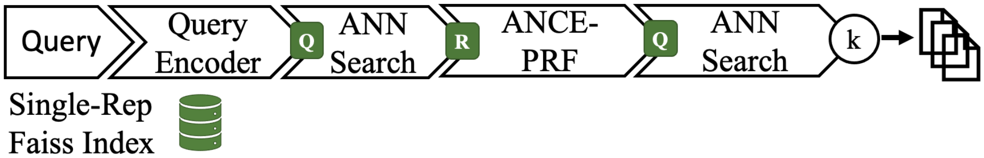


Figure 2.15: Conduct ANCE-PRF using PyTerrier.

Vector-PRF: Besides the *text-based* Dense PRF methods, i.e. ANCE-PRF, where PRF passages are appended to the original query to from the new reformulated query, a further *single representation embedding-based* Dense PRF method was proposed by Li et al. (2021a). Similar to ANCE-PRF, the Vector-PRF model also works within the single representation dense retrieval paradigm and consists of the three-stage PRF implementation pipeline, which we have introduced at the beginning of Section 2.4. In contrast to ANCE-PRF, Vector-PRF is unsupervised in nature and refines the query representation heuristically. By simply applying the average pooling (referred to as Vector-PRF) or a weighted interpolation (referred to as Rocchio-PRF) on the original query embedding and the pseudo-relevance feedback embeddings, Vector-PRF produces a more effective refined query representation. More specifically, the PRF stage of Vector-PRF can be described as follows: Vector-PRF obtains the top-k ranked passage embedding from the initial retrieved results as the PRF embeddings, denoted as $\psi_{p^1} \cdots \psi_{p^k}$, then the new query representation obtained using the average pooling method is as follows:

$$\psi'_q = \texttt{AvgPool}(\psi_q, \psi_{p^1} \cdots \psi_{p^k}). \tag{2.36}$$

In addition, the new query representation can be obtained using the weighted combination method as follows:

$$\psi'_q = \alpha_{Rocchio-PRF} \cdot \psi_q + \beta_{Rocchio-PRF} \cdot \texttt{AvgPool}(\psi_{p^1} \cdots \psi_{p^k}), \tag{2.37}$$

where the hyperparameters $\alpha_{Rocchio-PRF}$ and $\beta_{Rocchio-PRF}$ control the weights assigned for the original query and the PRF passages, respectively.

Overall, compared to ANCE-PRF, the Vector-PRF method is simpler to implement, however, the effectiveness of the Vector-PRF method has been observed to have a limited increase in terms of retrieval effectiveness. In this thesis, we employ both the ANCE-PRF and Vector-PRF models as baselines.

To summarise, we have argued that each type of Pseudo-Relevance Feedback (PRF) technique has its own strengths and drawbacks. More specifically, sparse PRF approaches (cf. Section 2.4.1), such as the RM3 model, are easy to implement and straightforward to interpret. The expansion terms added to the original query can effectively alleviate the lexical gap between the query and the document. However, these sparse PRF methods rely on lexical retrieval and thus struggle to address the semantic gap between the query and document. In contrast, neural PRF techniques (cf. Section 2.4.2), especially those based on the BERT model, can capture the deeper semantic relationships between words during the PRF stage, thereby alleviating the semantic mismatch problem. However, neural PRF techniques, such as CEQE and BERT-QE approaches, operate within the retrieval-and-rerank paradigm. This means that these neural PRF techniques still rely on the sparse index, and fall short when it comes to increasing recall performance, as the contextual meaning of an expansion term may be lost. Finally, ANCE-PRF and Vector-PRF approaches (cf. Section 2.4.3) perform the query expansion and retrieval using the dense index, thus promising to alleviate both semantic and lexical mismatch problems. However, as these dense-PRF techniques implement the pseudo-relevance feedback mechanism on the single presentation dense retrieval, interpreting the actual expanded embeddings becomes challenging.

In the following, we summarise the overall landscape of pseudo-relevance feedback and other related approaches in the literature in Table 2.3. From the table, it is evident that there exists a significant gap in the existing literature, as there have been no attempts to incorporate the pseudo-relevance feedback mechanism within the framework of a *multiple representation* dense index. In this work, we propose novel dense query expansion approaches for multiple representation dense retrieval to fill in this gap. Indeed, we find that the easy access to the token-level document embeddings used by the multiple representation dense retrieval paradigm, namely ColBERT (cf. Section 2.3.2) provides an excellent basis for our dense retrieval pseudo-relevance feedback approach. Indeed, while the use of embeddings in ColBERT addresses the vocabulary mismatch problem, we argue that identifying more related embeddings from the top-ranked documents may help to further refine the document ranking. In particular, as we will show, this permits representative embeddings from a set of pseudo-relevance documents to be used to refine the original query representation.

2.5 Retrieval Evaluation

To quantify how well a system is able to retrieve relevant information in response to a user's query, it is crucial to evaluate the *effectiveness* and the *efficiency* of a search engine. In particular, effectiveness performance assesses the ability of a search system to retrieve the

Table 2.3: Representative pseudo-relevance feedback and related approaches in the literature, organised into two dimensions: the task of the approach and the type of index it conducted on. L-Sparse denotes the Learned Sparse models, N-PRF denotes the Neural-PRF approaches and D-PRF denotes the Dense-PRF techniques.

Techniques		Index				Task		
		Sparse	Learned Sparse	Dense Single	Dense Multiple	Query Ref*	Document Ref*	Reranking
Sparse PRF	Rocchio (Rocchio, 1971)	✓				✓		
	Bo1 Amati and Van Rijsbergen (2002)	✓				✓		
	RM3 (Abdul-Jaleel et al., 2004)	✓				✓		
	Sparse External Expansion (Peng et al., 2009a)	✓				✓		
	CEQE (Naseri et al., 2021)	✓				✓		
	PGT (Yu et al., 2021a)	✓				✓		
Doc. Exp.	Doc2query (Nogueira et al., 2019a)	✓					✓	
	DocT5query (Nogueira et al., 2019b)	✓					✓	
	DeepCT (Dai and Callan, 2020a)	✓				✓	✓	
	HDCT (Dai and Callan, 2020a)	✓				✓	✓	
L-Sparse	SparTerm (Bai et al., 2020)		✓			✓	✓	
	EPIC (MacAvaney et al., 2020b)		✓			✓	✓	
	SPLADE (Formal et al., 2021a)		✓			✓	✓	
	DeepImpact (Mallia et al., 2021)			✓		✓	✓	
	COIL (Gao et al., 2021a)		✓				✓	
	UniCOIL (Lin and Ma, 2021)		✓				✓	
N-PRF	NPRF (Li et al., 2018)	✓						✓
	BERT-QE (Zheng et al., 2020)	✓						✓
D-PRF	ANCE-PRF (Yu et al., 2021b)			✓		✓		
	Vector-PRF (Li et al., 2021a)			✓		✓		

relevant documents to the top-ranked position in response to the users' information needs while efficiency concerns less response time to return the retrieved document list to the user. In the era of neural IR, many retrieval systems, such as learned sparse retrieval models (cf. Section 2.2.3) and dense retrieval models (cf. Section 2.3), need to be trained using some training dataset to learn to understand and represent the context and semantic relationship between different words. Therefore, training on the datasets addressing their retrieval tasks enables neural retrieval models to retrieve more relevant documents based on semantic similarity rather than simple keyword matching. In addition, performing the evaluation on the test queries that belong to the same domain as the training queries, which is also referred to as an *in-domain* dataset, it is also worth evaluating the trained retrieval models' performance on *out-of-domain* datasets.

Besides the quantitative analysis of the proposed methods, we also provide the qualitative analysis for each model. In particular, example queries for the qualitative analysis are selected based on the per-query retrieval effectiveness analysis between the proposed method and the compared baseline model. The most improved queries and degraded queries are selected as the representative examples.

In this section, we first introduce the various evaluation datasets used in this thesis (cf. Section 2.5.1). Then we introduce the most prominent methodologies for adhoc retrieval evaluation in Section 2.5.2. After this, we detail the different evaluation benchmarks in the era of neural IR in Section 2.5.3, namely in-domain evaluation and out-of-domain evaluation.

2.5.1 Evaluation Datasets

In this section, we describe the training datasets and test collections used in this work. In particular, neural IR models, like ColBERT (cf. Section 2.3.2), are trained using supervised data, specifically using the MSMARCO (Nguyen et al., 2016) passage dataset. Therefore, we introduce the popular training dataset for neural IR models. Moreover, to evaluate the performance of the information retrieval system in a controlled manner, a test query set is used, where the test queries are accompanied by relevance judgements. More specifically, Table 2.4 lists the statistics of the training corpora and the test query sets used in this thesis, where $|C|$ and $|Q|$ denote the document collection size and test query set size, respectively. $\overline{L(Q)}$ denotes the average query length in a query set while $\overline{Rel/Q}$ denotes the average number of relevant documents for the queries.

MSMARCO Passage Ranking Dataset: The name of the MSMARCO dataset is short for the large-scale MAchine Reading COmprehension dataset, which was released by Bing (Nguyen et al., 2016) in 2016. The dataset allows researchers to study the question answering and the adhoc ranking tasks, where a ranked list of the documents can be retrieved to potentially answer a given question. In particular, there are two ranking tasks: passage ranking and document ranking tasks. More specifically, the passage ranking dataset contains 8.8 million passage-length extracted from the document web pages. For each training in the passage ranking collection, there are on average 1.06 judged relevant passages. In order to train effective retrieval models, which are expected to distinguish the relevant passages from the non-relevant passages, the training instances are required to consist of positive (relevant) as well as negative (non-relevant) passages. Therefore, a triplet training dataset was curated for neural retrieval model training. In particular, for each query in the training dataset, a negative passage is randomly selected from the BM25 retrieved passages that have not been judged by human annotators.

Moreover, as discussed in Chapter 1, we focus on effective retrieval based on various neural pseudo-relevance feedback techniques. However, pseudo-relevance feedback approaches are known to be not effective on test collections with few judged passages (Amati et al., 2004), on average 1.07 judged relevant passages for each development query. Therefore, in this work, we do not conduct the evaluation using the MSMARCO Dev set.

There are 6838 test queries but their relevance judgements are not publicly available. An official MSMARCO passage ranking leaderboard[3] provides the ranked submissions based on their evaluation performance on the test queries. The official evaluation metric used is MRR@10.

MSMARCO Document Ranking Dataset: Besides the MSMARCO passage ranking dataset, the task organisers provide a counterpart longer document ranking dataset. In particular, the document ranking dataset comprises 3.2M web pages with various field information, namely URL, title and body text. There are several connections between the document ranking task to the passage ranking task. Firstly, the documents in the document ranking task contain the source pages of the passages from the passage ranking datasets. Secondly, the relevance judgements

[3] microsoft.github.io/MSMARCO-Passage-Ranking-Submissions/leaderboard/

of the document dataset are transferred from the passage judgements, which means a relevant passage occurring in the document indicates the document is also relevant to a query. Similar to the MSMARCO passage leaderboard, there is also a leaderboard[4] for document ranking with an official metric of MRR@100.

TREC Deep Learning Track 2019 & 2020: The TREC DL tracks have been organised by NIST and allow researchers from different institutions to evaluate their methodologies based on the released test topics. As discussed earlier, MSMARCO development queries have sparse judgements, hence the TREC track organisers release more densely judged test topics for both passage and document ranking tasks. In particular, TREC DL 2019 & 2020 topics are released based on the former introduced official MSMARCO (also referred to as MSMARCO v1) training data while TREC 2021 & 2022 topics are based on a larger-sized MSMARCO (often referred to as MSMARCO v2) dataset, where nearly 16 times increase in the size of the passage collection and nearly four times increase in the document collection size (Craswell et al., 2023, 2021b). The neural retrieval models are mostly trained using MSMARCO v1 dataset, therefore, in this thesis, we only evaluate using TREC 2019 (Craswell et al., 2020) and TREC 2020 (Craswell et al., 2021a) queries. More specifically, TREC 2019 Deep Learning track topics (43 topics with an average of 215.35 relevance judgements per query) and the TREC 2020 Deep Learning track topics (54 topics with an average of 210.85 relevance judgements per query) from TREC DL passage ranking task. In addition, for the document ranking task, there are 43 test queries from the TREC Deep Learning Track 2019 and 45 test queries from the TREC Deep Learning Track 2020 with an average of 153.4 and 39.26 relevant documents per query, respectively. The official evaluation metric for both TREC DL passage and document ranking is nDCG@10.

TREC Robust04 Dataset: The Robust04 collection contains 528K newswire articles from TREC disks 4 & 5 and it is proposed to address the adhoc retrieval task (Voorhees, 2004). The older Robust04 collection has made significant contributions to the retrieval community as it provides a large number of test topics with various types, namely 250 topics with 'title', 'description' and 'narrative' fields and highly complete pools with lots of relevant documents (cf. Table 2.3). In this thesis, we also conduct the evaluation using 250 title-only and description-only query sets from the TREC Robust04 document ranking task.

TREC GOV2 Dataset: The GOV2 corpus was used by the TREC Terabyte Tracks at TREC 2004–2006 and which addresses the web retrieval task. The GOV2 dataset comprises 52.3M web pages and was curated by the University of Glasgow. In this thesis, we also evaluate using 149 description-only queries from GOV2 (Clarke et al., 2004).

The WT10G Dataset: The WT10G (Web Track 10Gigabytes) dataset is also commonly used by researchers in the IR field, which is distributed by CSIRO in Australia. The WT10G collection is a subset of a more enormous collection, VLC2, and is used to perform adhoc retrieval. In this thesis, we also perform the evaluation using the 100 topics from the WT10G dataset.

[4] `microsoft.github.io/MSMARCO-Document-Ranking-Submissions/leaderboard/`

Table 2.4: Summary statistics for the data used in this thesis.

Dataset	**Tasks**	|*C*|	|*Q*|	$\overline{L(Q)}$.	$\overline{Rel/Q}$
		Training Data			
MSMARCO Passage	Web Retrieval	8,841,823	-	-	-
MSMARCO Document	Web Retrieval	3,213,835	-	-	-
Robust04 corpus (TREC disks 4&5)	News Retrieval	528,155	-	-	-
GOV2	Web Retrieval	25,205,179	-	-	-
WT10G	Web Retrieval	1,692,096	-	-	-
		Evaluation Data			
MSMARCO Passage Training	Web Retrieval	-	502,939	6.06	1.06
MSMARCO Passage Development	Web Retrieval	-	6,980	5.92	1.07
MMSARCO Passage Test	Web Retrieval		6,837	5.85	-
TREC DL 2019 Passage	Web Retrieval	-	43	5.40	215.34
TREC DL 2020 Passage	Web Retrieval	-	54	6.04	210.85
TREC DL 2019 Document	Web Retrieval	-	43	5.51	153.42
TREC DL 2020 Document	Web Retrieval	-	45	6.31	39.26
Robust04 (title)	News Retrieval	-	250	2.76	69.92
Robust04 (desc.)	News Retrieval	-	250	15.61	69.92
GOV2 [5]	Web Retrieval	-	149	11.62	180.65
WT10G (title)[6]	Web Retrieval	-	100	4.23	59.80
WT10G (desc.)[7]	Web Retrieval	-	100	11.62	59.80
DBPedia	Entity Retrieval	-	400	5.39	38.2
NFCorpus	Bio-Medical Information Retrieval	-	50	5.96	38.2
TREC-COVID	Bio-Medical Information Retrieval	-	50	10.60	493.2
Touché-2020	Argument Retrieval	-	49	5.39	19.0

BEIR Benchmark: In 2021, Thakur et al. (2021) constructed the BEIR benchmark which contains diverse retrieval tasks from different domains. Most of the dense retrieval models introduced in Section 2.3 are trained using the MSMARCO (v1) corpora. However, these dense retrieval models might be ineffective when evaluating using topics from other domains. Therefore, the BEIR datasets are popularly used to evaluate the out-of-domain retrieval performance, which we further detail in Section 2.5.3. As pseudo-relevance feedback techniques are known to be not effective on test collections with few judged documents (Amati et al., 2004), we only evaluate the BEIR datasets that have a good number of judgements for each query. Hence, in this work, we evaluate on four BEIR (Thakur et al., 2021) datasets, namely DBPedia (Hasibi et al., 2017), NFCorpus (Boteva et al., 2016), TREC-COVID (Voorhees et al., 2021) and Touché-2020 (Bondarenko et al., 2020).

2.5.2 Evaluation Metrics

In this section, we discuss the fundamental concepts of both the effectiveness and efficiency evaluation metrics, their importance, and the used metrics in this work.

Effectiveness: Given a query q, a retrieval system returns a ranked list R_q by searching over the index. In practice, instead of returning all the retrieved results, the top k returned documents $R_k(q)$ are selected to return to the user. In the test query set, among the top k returned documents in response for a given q, the documents that can satisfy the user's information need are judged as relevant and denoted as $Rel(q)$. In particular, the relevant documents being retrieved can be denoted as the intersection of the two sets $Rel(q)$ and $R_k(q)$.

The two most basic effectiveness-associated metrics are *Precision* and *Recall* (Cleverdon et al., 1966). In particular, precision measures the fraction of the retrieved documents that are relevant to the input query with k cutoff, which can be calculated as follows:

$$P(q,k) = \frac{|Rel(q) \cap R_k(q)|}{|R_k(q)|} \tag{2.38}$$

Moreover, Recall is an evaluation metric that measures the proportion of relevant documents retrieved among the total number of relevant documents in the collection. Recall is important for IR systems that prioritise retrieving a larger portion of relevant documents, even at the cost of retrieving some irrelevant documents (cf. decreased precision). Recall at cutoff k is calculated as:

$$R(q,k) = \frac{|Rel(q) \cap R_k(q)|}{|Rel(q)|} \tag{2.39}$$

However, both precision and recall metrics are set-based metrics and therefore they are insensitive to the ranking swaps in the final ranking list (Robertson, 2008). Therefore, *Average Prevision* (AP) metric (Harman, 1995), which takes the order into consideration, is popularly used to address this limitation. Average Precision is defined as:

$$AP(q,k) = \frac{\sum_{i=1}^{k} \mathrm{P}(q,i) Rel_i}{|Rel(q)|}, \tag{2.40}$$

where $P(q,i)$ denotes the precision calculated using Equation (2.38) and $[Rel_i]$ is the relevance judgement, e.g. 0 for non-relevant and 1 for relevant, of the i-th document is relevant to the input query.

Instead of only assessing the retrieval effectiveness for a single input query, in the more realistic scenario, the assessment of the retrieval system's effectiveness must be conducted on a collection of query sets Q with various query types. Based on average precision, one of the most commonly used metrics is *Mean Average Precision* (MAP) (Craswell and Hawking, 2004, Craswell et al., 2003), which is simply calculated by averaging over the average precision value for each query q in the query set Q.

$$MAP(Q,k) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} AP(q_j,k) \tag{2.41}$$

The MAP metric summarises the retrieval performance of the rankings from multiple queries. One assumption of average precision (AP and MAP) is that the relevance of a document to an input query is binary. However, in large-scale web search scenarios, the level of the relevance can vary for different documents. Indeed, in some cases, relevant information may only exist in a paragraph of a long document. Therefore, the relevance of a document should be assessed using a graded scale, from non-relevant to highly relevant. Moreover, the position of the graded relevance also matters, as users expect to read the highly relevant documents in the top position while tend to neglect the lower ranked documents. Accordingly, Järvelin and Kekäläinen (2002) proposed *discounted cumulative gain* (DCG) metric, which includes a logarithmic discount factor to assign higher weights to the relevance of documents that are ranked later in the list and lower weights to documents that are ranked higher. DCG encodes two assumptions (Järvelin and Kekäläinen, 2002):

- Highly relevant documents are more useful than marginally relevant documents
- The lower the ranked position of a relevant document, the less useful it is for the user, since it is less likely to be examined.

The DCG for a query q and given rank cutoff k can be calculated as follows:

$$DCG(q,k) = \sum_{i=1}^{k} \frac{2^{Rel_i} - 1}{\log_2(i+1)}, \tag{2.42}$$

where Rel_i denotes the graded-relevance of the i-th document in the ranking list. The logarithmic function log_2 is the discount factor to reduce the contribution of the lower ranked documents.

An issue arises with DCG when we compare the performance for different queries as the relevance and the length of the result list depend on the specific queries entered by users. The normalised version of DCG (nDCG) is often used to address this issue, where a normalised factor enables the comparison of the effectiveness of different systems across different queries (Järvelin and Kekäläinen, 2002).

$$nDCG(Q,k) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{DCG(q_j,k)}{IDCG(q_j,k)}, \tag{2.43}$$

where $IDCG(q,k)$ is an ideally perfect ranking of a query q and is calculated as follows,

$$IDCG(q,k) = \sum_{i=1}^{|REL_k|} \frac{2^{Rel_i} - 1}{\log_2(i+1)}, \tag{2.44}$$

where REL_k represents the list of relevant documents (ordered by their relevance) in the corpus up to the rank cutoff k.

Although DCG incorporates a discount factor, Craswell et al. (2008) noted that it does not address the ‘click position bias’, where users are more likely to click on higher-ranked documents than those lower-ranked ones. Furthermore, DCG’s potential is constrained in situations where there is only a single relevant document for a query, such as when one searches for ‘homepage of the University of Glasgow'. The Reciprocal Rank (RR) metric, as introduced by Collins-Thompson et al. (2014), is often employed for such cases. RR is defined as the reciprocal of the rank of the first relevant document retrieved. In practice, for a query set Q, *Mean Reciprocal Rank* (MRR) is often used by averaging the reciprocal ranks over the queries in a set Q. The MRR metric can be calculated as follows:

$$MRR = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{\text{rank}_{q_j}}, \tag{2.45}$$

where rank_{q_j} denotes the rank of the first relevant document for the j-th query q_j in the test query set Q. As discussed in Section 2.5.1, MRR@10 is the official evaluation metric used in the MSMARCO passage and document leaderboard. In this work, we continue to report models' MRR@10 performance. However, MRR is suitable for test queries with binary relevance scenarios and may not be well extended to measure the effectiveness for queries that have different levels of relevance. Therefore, we are more focused on MAP and nDCG metrics when comparing among baselines.

Efficiency: Different to the effectiveness metrics, the efficiency metrics quantify the speed and resource usage of a retrieval system, such as the query throughput and query latency as well as the indexing time and index storage requirements (Frachtenberg, 2009, Tonellotto et al., 2018). From the angle of the users, query latency, which measures how long the users will have to wait for a response, is a critical metric as it directly impacts the user's search experience. If a retrieval system takes too long to respond to the input queries, users will be impatient and resort to a faster retrieval system (Arapakis et al., 2014). For a set of queries, a common way is to measure the mean response time or the mean execution time (measured in milliseconds) that a retrieval system completes the retrieval for all the queries.

On the other hand, the cost of indexing and its storage should be also taken into consideration. Especially in the era of neural IR, in addition to the traditional sparse index, various index structures have been proposed, for instance, an approximate nearest neighbour search index, where each token or document in the collection is encoded and indexed in the dense index, e.g. the FAISS (Johnson et al., 2019) index. Therefore, it is important that we measure the storage requirements of the index structures.

2.5.3 Evaluation Benchmarks

As we detailed in Section 2.5.1, various dense retrieval models do not generalise well to user queries from other domains. Two primary approaches have emerged for the evaluation of the neural-based retrieval models: in-domain and out-of-domain evaluations. These methods are distinguished by the distribution differences between the training data and the test data.

In particular, for in-domain evaluation, the test data are similar to the data used to train the model. For instance, a ColBERT (Khattab and Zaharia, 2020) model is trained using the MSMARCO (Nguyen et al., 2016) training dataset and then evaluated using the MSMARCO development query set. This constitutes an example of in-domain evaluation. In-domain evaluation can be used to assess how effectively the model has learned from the task-specific data. In contrast, the evaluation conducted using queries and documents from a domain different from the one the system was designed for or trained on is commonly referred to as out-of-domain (or *out-of-distribution* and *zero-shot*) evaluation. For example, we might train a ColBERT dense retrieval model on web-search task-related queries and then test its performance on news retrieval task-related queries (Santhanam et al., 2022). Evaluating out-of-distribution performance for neural retrieval models is important, as it measures the model's generalisation capabilities from the trained retrieval to other domains.

To summarise, in this section, we first introduce the effectiveness and efficiency metrics in Section 2.5.2. It is worth noting both effectiveness and efficiency are important for an IR system, as it ensures that users promptly receive highly relevant results while keeping operational costs low. (Asadi and Lin, 2013, Manning, 2009, Tonellotto et al., 2013, 2018). Ideally, a good IR system should be both effective and efficient: it should return highly relevant results with minimal response time. In this thesis, we empirically evaluated the effectiveness of various retrieval models using MAP, Recall, nDCG and MRR. More specifically, the MAP and Recall metrics can reflect the success of the initial retrieval within the multi-stage retrieval pipeline, where sufficient high-quality candidate documents should be provided to the reranker model. Unless otherwise specified, we calculate MAP and Recall at rank 1000. In addition, we report MRR and nDCG calculated at rank 10. In particular, MRR@10 is employed as the official metric to compare the effectiveness of various submitted systems in the MSMARCO leaderboard (Nguyen et al., 2016),[8] while nDCG@10 is the official effectiveness metric used by TREC Deep Learning Track (Craswell et al., 2021a, 2020, 2021b). Moreover, in Section 2.5.3, we clarified the concepts of different evaluation benchmarks for IR. We note that, in the era of neural IR, both in-domain and out-of-domain evaluations are crucial. In-domain evaluation helps fine-tune the retrieval system's performance on its intended data, while out-of-domain evaluation encourages the retrieval system to be more robust across various domains. In this thesis, the proposed neural-based models are mainly trained based on the publicly available MSMARCO datasets, which contain a collection of datasets focuses on deep learning in search.[9] The MSMARCO dataset is widely used to train neural retrieval models, as it is the only available dataset with large numbers of training queries. However, the development set of the official MSMARCO dataset is sparsely judged, with an average 1.1 judgment rate. Accordingly, the TREC Deep Learning Track provides more densely judged query sets. Therefore, for in-domain evaluation, we mainly evaluate using the TREC query sets, namely the TREC 2019 and TREC 2020 queries. For out-of-domain evaluation, Thakur et al. (2021) released 18 publicly available datasets for various tasks, in addition to the

[8] `microsoft.github.io/MSMARCO-Document-Ranking-Submissions/leaderboard/`

[9] `microsoft.github.io/msmarco/`

MSMARCO retrieval, out-of-domain text retrieval tasks known as BEIR. Therefore, we also evaluate our proposed techniques in terms of their out-of-domain performance using four of the BEIR datasets that have dense juedgements.

2.6 Conclusion

This chapter provides comprehensive background knowledge, as well as related works, from traditional sparse retrieval to neural-based retrieval methods and up-to-date dense retrieval techniques for adhoc information retrieval.

In particular, we first introduce the classical sparse retrieval paradigm and various prevalent term weighting models together with how we construct the retrieval pipelines for experiments in Section 2.1. Although efficient, sparse retrieval models can fail the retrieval due to the lexical or semantic mismatch problem between the user query and collection. Therefore, in Section 2.2, we introduced the more advanced contextualised pretrained language models, PLMs, e.g. BERT, followed by their advanced implementations in IR, namely the multi-stage retrieval and the document representation refinement. Moreover, we introduced the more recently proposed dense retrieval paradigms in Section 2.3, which operate entirely based on dense representations of the query and document. In particular, dense retrieval models can capture the contextualised matching relationship between queries and documents, hence to an extent alleviating the semantic mismatch issue. In addition, pseudo-relevance feedback mechanisms can lift the effectiveness performance by overcoming the lexical mismatch problem. Therefore, we detailed various PRF techniques, from the sparse PRF approaches to the neural PRF approaches in Section 2.4. However, on one hand, we found that existing neural-PRF techniques are primarily based on the BERT PLM. The potential of Sequence-to-Sequence PLMs, such as T5, for query reformulation has not been sufficiently explored. On the other hand, we dscovered that the retrieval can still fail when the user's queries are underrepresented even within the dense retrieval paradigm.

In particular, this thesis focuses on implementing pseudo-relevance feedback techniques to perform query reformulation for both sparse retrieval and dense retrieval paradigms. More specifically, in Chapter 3, we study the effectiveness of a Sequence-to-Sequence query reformulation method for sparse retrieval and the multi-stage retrieval pipelines. Next, in Chapter 4, we introduce our proposed ColBERT-PRF model, which implements the pseudo-relevance feedback mechanism entirely within the multiple-representation dense retrieval paradigm to refine the query representation. In addition, Chapter 5 explores the effectiveness of external pseudo-relevance feedback techniques for out-of-domain retrieval tasks. Next, in Chapter 6, we extend ColBERT-PRF to Col⋆-PRF with various underlying pretrained language models. Finally, we propose a deep language model-based feedback weighting model, called CWPRF in Chapter 7.

Chapter 3

Generative Query Reformulation for Effective Adhoc Search

In our thesis statement (cf. Section 1.1), we postulated that pseudo-relevance feedback information can be used by a Sequence-to-Sequence neural model to generate more effective query reformulations for sparse retrieval (cf. Section 2.1). In Chapter 1, we mentioned that discrepancies between the way that users express their information needs and the content of relevant documents can cause such documents not to be retrieved or highly ranked, thus causing issues such as semantic or lexical mismatch. Several threads of research have tried to bridge the pervasive query-document mismatch problem in information retrieval namely dense embedding-based representations, document expansion and query expansion.

Indeed, for the first thread, as introduced in Section 2.2, recent advances in pretrained neural network approaches have been shown to improve various text-processing tasks and to have the ability to learn the semantic meaning of the input text. Indeed, various approaches have demonstrated that the BERT-based deep neural ranking models, examples of which were introduced in Section 2.2.2, are capable of capturing the semantic and syntactic relationship between the texts, thus can mitigate the semantic mismatch between query and document. However, apart from a thread of work in dense retrieval models introduced in Section 2.3, which struggle to handle long documents and suffer from limited interpretability, many works have employed BERT models as neural *rerankers*, in that they are applied to improve an initial ranking obtained from an inverted index using models such as BM25. In addition, industry has started performing BERT-based reranking at scale (Wang et al., 2021b), demonstrating the value and applicability of the retrieve-then-rerank (cf. Section 2.2.2) paradigm. Therefore, in this chapter, we work within the retrieve-then-rerank framework and investigate the capability of using the Sequence-to-Sequence model for more effective query reformulations.

To overcome the lexical mismatch problem, approaches such as query expansion (cf. Section 2.4.1) and document expansion (cf. Section 2.2.3) are still promising, in order to create a candidate set of documents with a sufficiently high recall to enable a neural reranker to identify and up-rank relevant documents occurring in the candidate set. As introduced in Section 2.2.3, document expansion augments each document with additional information, for instance, additional terms selected from a corpus (Billerbeck and Zobel, 2005, Dai and Callan, 2020a) or predicted queries generated by a natural language generation model from the original document. Indeed, as argued in Section 2.2.3, the so-called doc2query approach expands each document by generating multiple predicted queries using generative language models before indexing. This is effective but comes with the significant upfront cost of applying a heavy model to every document at indexing.

On the other hand, as introduced in Section 2.4, query expansion typically describes approaches that modify the users' query with the aim to improve the retrieval effectiveness, such as the RM3 and Bo1 techniques introduced in Section 2.4.1. By adding additional terms selected from the pseudo-relevant set of returned documents in response to the initial query or by expanding the original query with lexical-level (Zukerman and Raskutti, 2002) or phrase-level (Riezler et al., 2007) paraphrases, the distance between the (reformulated) query and the relevant document(s) is reduced. Going beyond the traditional sparse PRF approaches (cf. Section 2.4.1), neural-based PRF approaches (cf. Section 2.4.2), such as CEQE, identifies expansion terms from the pseudo-relevant documents in the BERT embedding space. These approaches are compatible with the neural rerankers that were introduced in Section 2.2.2, in that a refined ranking list obtained using query expansion can improve the effectiveness of a BERT-based reranking model. Furthermore, as introduced in Section 2.4.2, neural pseudo-relevance feedback (PRF) models have been proposed, such as NPRF and BERT-QE. However, these models are limited in their functionality, in that the additional relevance signal obtained from the pseudo-relevant set is only used to re-rank the candidate set of documents, rather than creating a higher recall candidate set by re-executing a reformulated query on the inverted index. Instead, in this chapter, we aim to investigate the potential of generative neural models to produce a refined candidate set of documents by generating a refined query reformulation for a sparse retrieval approach.

Indeed, motivated by the fact that the same information need can be formulated using different natural language expressions, in this chapter, we focus on expanding the original query by generating *paraphrases* that share the same information need, to improve the retrieval effectiveness. We cast the query reformulation task as a text generation task. This allows us to be able to test whether the knowledge encapsulated by *pretrained* text generation models — such as T5 (cf. Section 2.2.1) and a more advanced instruction fine-tuned T5 variant called FLAN-T5 (Wei et al., 2021) — can be exploited for query reformulation to provide as training examples. We explore two possible generative query reformulation frameworks, GenQR and GenPRF. The first produces reformulations using only the user's query text itself (GenQR). The second approach makes use of pseudo-relevant documents as contextual information (GenPRF), to guide the reformulation process. For T5, we explore techniques to fine-tune the model for each query reformulation task. One challenge we found when fine-tuning the T5 model for query reformulation is that there are no gold-standard labelled query reformulations. To circumvent the lack of ground-truth data, we investigate the use of weakly supervised query pairs to fine-tune the T5 model, instead of human-annotated query pairs. In particular, we leverage three different filters that improve the weakly supervised dataset to reduce the noise present in the query pairs and thus reduce the training time of GenQR and GenPRF when injecting the task-specific knowledge into the T5 model. Meanwhile, for FLAN-T5, we explore prompting techniques in an attempt to leverage the patterns observed in the pre-training process.

In summary, this chapter makes four contributions:

- We propose two generative query reformulation frameworks, namely GenQR and GenPRF;
- We instantiate our generative frameworks using two generative models, a pretrained language model, T5, and an instruction fine-tuned language model, FLAN-T5, to reformulate input queries;
- We make use of pseudo-relevance feedback information as the contextual information for the input of the generative model;
- We demonstrate the effectiveness of the generated query reformulations in comparison to several existing baselines within the prevalent retrieve-then-rerank pipeline;
- We investigate the effectiveness of the neural reranker, the monoT5 reranker (Nogueira et al., 2020), when combined with GenQR and GenPRF results.

The remainder of this chapter is organised as follows: Section 3.1 presents our proposed generative query reformulation frameworks. Research questions and the used experimental setup are detailed in Sections 3.2 & 3.3, respectively. Next, we report and discuss our experimental results in Section 3.4. Finally, we summarise our findings and provide future work directions in Section 3.5.

3.1 Generative Query Reformulation

In Section 2.4, we introduced two possible query reformulation processes. More specially, depending on the input with or without pseudo-relevant feedback information, there are two scenarios: $\mathcal{P}_{QR}$, for which input is only the original query, and $\mathcal{P}_{PRF}$, where input is the original query and the pseudo-relevance feedback set of documents. In both scenarios, $\mathcal{P}_{QR}$ and $\mathcal{P}_{PRF}$ can be seen as methods that add candidate terms or phrases to rewrite or expand the original query (with the optional weighting of such terms). Instead, we rely on a T5 text generation model for $\mathcal{P}$, which can refine the initial query q^0 by generating paraphrases of the original query. In addition, to ensure that the paraphrases of the original query are on-topic in relation to the initial query, we also study an approach that can also encapsulate pseudo-relevance information as context during the reformulation process. Thus, we study two generative query reformulation frameworks: one that takes the form of a general query reformulation process (cf. GenQR), along with a pseudo-relevance feedback-based query reformulation (cf. GenPRF). For each query reformulation paradigm, we introduce two methods, namely fine-tuning and prompting, to leverage the pretrained knowledge of large language models for query reformulation.

In the following, we first introduce the GenQR framework in Section 3.1.1. Then we describe our GenPRF framework in Section 3.1.2. Finally, the weak supervision data process used to fine-tune the proposed GenQR and GenPRF when performing the knowledge injection method, and how it is filtered for quality is introduced in Section 3.1.3.

3.1.1 GenQR

Fine-tuning: Formally, our generative query reformulation process via injecting the task-specific knowledge into a pretrained text generation model (a pretrained T5 model) is denoted as $\mathcal{P}_{T5QR}$. In particular, the $\mathcal{P}_{T5QR}$ takes the initial query q^0 (in the form of a sequence of text and optionally other information) and produces refined queries. When conditioned only on the input query $q^0 = q^0_1, ...q^0_{|q^0|}$, a query paraphrase can be generated by applying a fine-tuned text generation function, $T5()$, as follows:

$$\texttt{T5}(\text{“refine”}, q^0_1, ...q^0_{|q^0|}, \text{“</s>”}),$$

where in the input sequence, the input query is prepended with a special prompt token “refine” indicating to T5 that it should reformulate the input and it is suffixed by the special end-of-sequence token “</s>”.

More specifically, as introduced in Section 2.2.1, T5 is a large pretrained Text-To-Text-Transfer-Transformer model. Within such a Sequence-to-Sequence framework, all the tasks can be cast in a Text-to-Text format. After pre-training, the pretrained knowledge is stored as the parameters of a T5 model, denoted as $T5(\bar{\theta})$. The T5 model consists of separate stacks of encoding and decoding layers: each encoding layer contains a self-attention layer and a feed-forward layer; while each decoding layer comprises a self-attention layer, an encoder-decoder attention layer and a feed-forward layer. The encoder takes a source sequence $X = (x_1, x_2, ..., x_n)$ with a length of n terms and the model is trained to produce the corresponding target sequence $Y = (y_1, y_2, ..., y_m)$ with a length of m terms. The ultimate goal when training a T5 model is, given a set of training pairs $\mathcal{M} = \{\langle X, Y \rangle\}$, for each pair $\langle X, Y \rangle$ to maximise the posterior for a target output Y given an input sequence X. The T5 decoder produces one token at each step, thus at each step during training, the maximum-likelihood objective function is:

$$\max P_{T5}(Y \mid X) = \max \prod_{t=1}^{M} P_{T5}(y_t \mid y_{1:t-1}, X, \bar{\theta}), \tag{3.1}$$

where $y_{1:t-1} = (y_1, \ldots, y_{t-1})$ are the tokens generated in the previous steps. At each step t, T5 maximises the conditional probability in Equation (3.1) by minimising the negative log probability

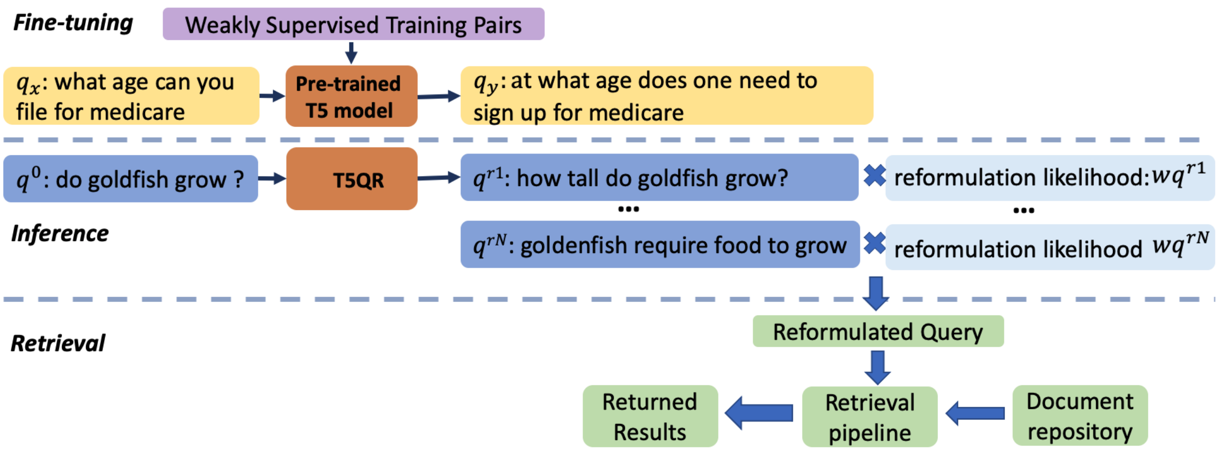

Fine-tuning
Weakly Supervised Training Pairs
q_x: what age can you file for medicare
Pre-trained T5 model
q_y: at what age does one need to sign up for medicare
q^0: do goldfish grow ?
T5QR
q^{r1}: how tall do goldfish grow?
reformulation likelihood: wq^{r1}
q^{rN}: goldenfish require food to grow
reformulation likelihood wq^{rN}
Inference
Retrieval
Reformulated Query
Returned Results
Retrieval pipeline
Document repository

Figure 3.1: The proposed T5QR model for fine-tuning a pretrained model for the adhoc query reformulation task.

as the prediction loss, also known as the Cross-Entropy (CE) loss:

$$\mathcal{L}_{T5}(y_t) = -\sum_{i=1}^{C} \tau_i \cdot \log(P_{T5}(c_i \mid y_{1:t-1}, X, \bar{\theta})), \tag{3.2}$$

where, $c_1, c_2, ..., c_C$ are the search space, i.e. the classes for the decoder – indeed, for a text generation task, C is equal to the length of the output vocabulary of the model; τ_i is an indicator variable equal to 1 when $y_t = c_i$ and 0 otherwise; and $P_{T5}(c_i|\cdot)$ is the predicted probability produced by the soft-max layer of the decoder. The loss in Equation (3.2) is backpropagated through the network to tune the model's parameters, such that the model's output is closer to the target.

T5 generates text *auto-regressively*, that is, the probability of a generated sequence is calculated as the product of the probability of each token in the sequence. By applying a fine-tuned T5QR model, $T5QR(\hat{\theta})$, the joint likelihood of the query reformulation is:

$$P\left(q_1^r, q_2^r, ..., q_{|q^r|}^r\right) = \prod_{t=1}^{|q^r|} P_{T5QR}(q_t^r | q_{1:t-1}^r, q^0, \hat{\theta}), \tag{3.3}$$

Hence, the final output of $\mathcal{P}_{T5QR}(q^0)$ is obtained by combining the output sequences of N applications of the T5 model, and weighting the terms from each sequence by the joint likelihood of the sequence, as follows:

$$\mathcal{P}_{T5QR}(q^0) = w_{q^{r1}} \cdot \left[q_1^{r1}, ...q_{|q^{r1}|}^{r1}\right] + ... + w_{q^{rN}} \cdot \left[q_1^{rN}, ...q_{|q^{rN}|}^{rN}\right], \tag{3.4}$$

where w_{q^r} denotes the joint likelihood of reformulation (or paraphrase of the input query) q^r – i.e. $w_{q^r} = P\left(q_1^r, q_2^r, ..., q_{|q^r|}^r\right)$ – and N is the number of predicted output sequences used to form a query reformulation in response to the original query q^0. By generating and combining N paraphrases of the original query generated by T5, important terms are more likely to receive higher

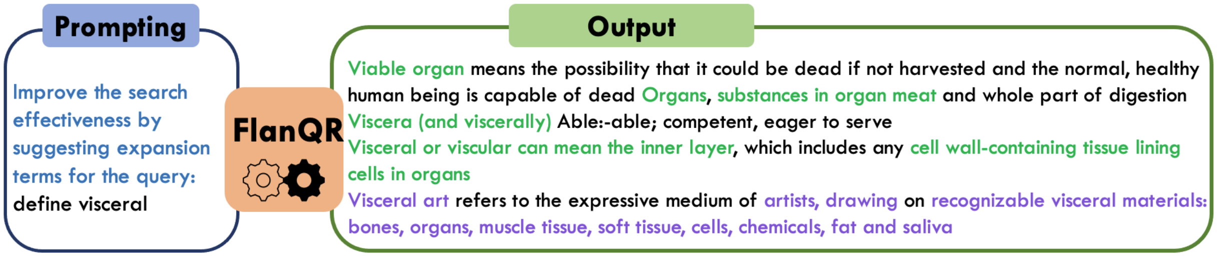

Prompting
Improve the search effectiveness by suggesting expansion terms for the query: define visceral
FlanQR
Output
Viable organ means the possibility that it could be dead if not harvested and the normal, healthy human being is capable of dead Organs, substances in organ meat and whole part of digestion
Viscera (and viscerally) Able:-able; competent, eager to serve
Visceral or viscular can mean the inner layer, which includes any cell wall-containing tissue lining cells in organs
Visceral art refers to the expressive medium of artists, drawing on recognizable visceral materials: bones, organs, muscle tissue, soft tissue, cells, chemicals, fat and saliva

Figure 3.2: The proposed FlanQR model for the adhoc query reformulation task.

weights – indeed, a similar repeated application of T5 is used by docT5query (Nogueira et al., 2019a). The fine-tuning process for T5PRF is similar to T5QR, the only difference is the input template. The final reformulated query for T5PRF is obtained by combining the weighted combinations of N invocations of Equation (3.6), in a similar manner as for T5QR in Equation (3.4).

Figure 3.1 illustrates the overall framework of the GenQR process. We first fine-tune a pretrained T5 model using our proposed QPP-based regularised loss. The obtained reformulated query concatenates to the original query as a new query to performance retrieval task. Each pair of training queries consists of a source query and a target query, where the source query is prepended with the word "paraphrase" and suffixed by the end-of-sequence token "$\langle /s \rangle$".

Prompting: Instead of injecting task-specific knowledge via fine-tuning a language model, FLAN-T5 employs an instruction-tuning approach to improve the ability of a language model for various tasks. However, the capability of FLAN-T5 to perform query reformulation tasks is still unclear. Thus, besides fine-tuning a T5 model, we also investigate employing the FLAN-T5 model for reformulating the initial user query and we denote this query reformulation process as $\mathcal{P}_{FlanQR}$. More specifically, we design a query reformulation task aware prompt and combine it with the original query q^0 as the input for $\mathcal{P}_{FlanQR}$. Our $\mathcal{P}_{FlanQR}$ model does not require any further training and only depends on the original query as well as the prompt. The input template for $\mathcal{P}_{FlanQR}$ is as follows:

$$\texttt{FlanQR}(<\texttt{Prompt}>, q_1^0, ...q_{|q^0|}^0). \tag{3.5}$$

Figure 3.2 illustrates the framework of the FlanQR process. However, while the user's input queries may be short and may not provide sufficient evidence to interpret the meaning of the input query. We hypothesise that the additional pseudo-relevant context information can aid the query reformulation process. In the next section, we show how T5QR and FlanQR can be formulated as pseudo-relevance feedback mechanisms.

3.1.2 GenPRF

To reinforce the capability of the query reformulation model to interpret the meaning of the input query, an additional context that further explains the query statement can be desirable. Motivated by the utility of the top returned documents in pseudo-relevance query expansion models (such as Bo1 and RM3 that were introduced in Section 2.4.1), we leverage the top returned documents for the original query and use them as the pseudo-relevance contextual information for the query. This allows the model to better interpret the user's search intent, thus avoiding the semantic mismatch problem.

Contextual Fine-tuning: The query together with its contextual information is taken as the input to fine-tune a pretrained T5 model:

$$\texttt{T5PRF}(\text{“refine”}, q^0_1, ...q^0_{|q^0|}, \text{“context:”}, context(R), \text{“</s>”}), \tag{3.6}$$

where (similar to “refine”) “context:” is a prompt token denoting the start of the context passage(s); $context(R)$ identifies important passage(s) of text from the pseudo-relevance feedback document set R. Thereafter, we instantiate this approach under the GenPRF framework as T5PRF.

Contextual Prompting: Additionally to T5PRF, we also instantiate the GenPRF as FlanPRF, where we direct prompt the FLAN-T5 model using a task-aware prompt, the initial query q^0 as well as the top-ranked pseudo-relevance feedback information $context(R)$ as input. Note that similar to $\mathcal{P}_{FlanQR}$, no fine-tuning is needed for $\mathcal{P}_{FlanPRF}$. The input template for $\mathcal{P}_{FlanPRF}$. can be expressed as follows,

$$\texttt{FlanPRF}(<\texttt{Prompt}>, q^0_1, ...q^0_{|q^0|}, context(R)). \tag{3.7}$$

Types of Contextual Information Rather than use whole documents, we use passages as pseudo-relevance context information. Indeed, two considerations prevent entire feedback documents from being used as context information. Indeed, the compute and memory requirements of the transformer architecture rise exponentially as sequence length increases, so lengthy documents can easily exceed available memory. Moreover, long documents may not be wholly concerned with the topic of the query, potentially misleading our PRF models in understanding the meaning of the input query. The whole process can be described as follows: Firstly, for a given initial query $q0$, we obtain the first K documents returned by the retrieval model, i.e. $R_K(q0)$; Secondly, a sliding window with size w is used to break each document into passages with an overlap of $w/2$ words between neighbouring passages – we denote the text passages obtained from document d as $passages(d)$; Thirdly, a ranking model is employed to assign a score for each passage, measuring the relevance of the content of the passage to the query; finally, we apply one of three selection mechanisms to obtain the context information for the T5PRF model, namely the FirstP, TopP and MaxP selection approaches.

The intuition behind FirstP is that the leading paragraph in a long document often contains the main gist of the document. Hence, we pick the M highest scoring among each of the first passages of all documents in R, as follows:

$$context_{FirstP}(R, q^0) = \underset{p \in passages(d)[0], \forall d \in R}{\arg\overset{M}{\max}\text{s}} \; s(p, q^0), \tag{3.8}$$

where $passages(d)[0]$ denotes the first passage obtained from document d, and $s(p, q^0)$ denotes the relevance score given by a ranking model between a passage p and the original query. $\arg\max\text{s}^M()$ is an extension of $\arg\max$, which returns the top M scoring items.

Next, rather than obtaining the first passages, we look to identify the most relevant passages from the feedback set R, inspired by the MaxPassage ranking approach (cf. Section 2.1.2). In particular, we formulate two selection mechanisms, namely TopP and MaxP, which slightly differ in how they obtain the highest M scoring passages from R. TopP takes the M highest scoring passages across all of the feedback set, while MaxP selects the highest scoring passages from each document and then selects the highest M among these:

$$context_{TopP}(R, q^0) = \underset{p \in passages(d), d \in R}{\arg\overset{M}{\max}\text{s}} \quad s(p, q^0) \tag{3.9}$$

$$context_{MaxP}(R, q^0) = \underset{d \in R}{\arg\overset{M}{\max}\text{s}} \quad \underset{p \in passages(d)}{\arg\max} \quad s(p, q^0). \tag{3.10}$$

3.1.3 Weakly Supervised Query Pairs and Filters

In order to fine-tune T5 for the T5QR and T5PRF models, we need a large number of training pairs that contain a labelling signal. However, for the adhoc query reformulation task, there is no readily-available large-scale labelled dataset of query reformulations. Hence, to circumvent this shortage, we leverage existing test collections to generate the required weak supervision signal for T5.

In particular, to allow T5 to learn how to generate a paraphrase in response to an input query, the training pairs should convey the same information need. In other words, they should be semantically similar to each other. Following the work of Zerveas et al. (2019), we first identify pairs of queries that share at least one relevant document from a test collection. This forms our initial pool. Our main underlying assumption is that if a document is labelled as relevant for multiple queries, such queries are assumed to convey the same information need (Wang et al., 2020b, Zerveas et al., 2019). However, a clear risk when directly using the initial training pool is that noise can arise – for instance, the content of a document might address multiple different topics, or the queries pertain to different information needs. Hence, to reduce this noise, we introduce three filters that can be applied individually – or in combination, – in order to further improve the quality of the initial training pool, as well as reduce training time.

More specifically, we first identify all pairs of queries that are associated to the same relevant document. Then we extract the pairs of queries from these tuples as the initial pool $\mathcal{M}$. Within each query pair $\langle q_x, q_y \rangle$ in $\mathcal{M}$, each query consists of a sequence of words, i.e. $q = (q_1, q_2, \cdots q_{|q|})$. To refine $\mathcal{M}$, we apply filters to reduce the number of query pairs, or refine those query pairs to provide more useful reformulations for fine-tuning T5, i.e. $\mathcal{M}' = \mathcal{W}(\mathcal{M})$. We now introduce our three proposed filters.

3.1.3.1 Overlap Filter

This filter assumes that only query pairs for which there is a marked overlap in the retrieved documents for both queries are suitable training examples. A similar approach was previously described by Cronen-Townsend et al. (2004) for identifying a query drift in query expansion. Similarly, Mass et al. (2020) employed this method to identify a potential topical drift when generating paraphrases of FAQ questions. In particular, we use an overlap-based filter – denoted as $\mathcal{W}_O$ – for selecting high-quality query pairs from the initial pool $\mathcal{M}$. For a pair of queries $\langle q_x, q_y \rangle \in \mathcal{M}$, we issue q_x and q_y against the corpus index, respectively, and check the number of shared documents in the returned top-K result lists. The overlap of the result lists is defined as the cardinality of the intersection of the top K documents of the two result lists for q_x and q_y:

$$O(\langle q_x, q_y \rangle) = |R_K(q_x) \cap R_K(q_y)|. \tag{3.11}$$

Hence, the overlap weak supervision filter is defined as:

$$\mathcal{W}_O(\mathcal{M}) = \{\langle q_x, q_y \rangle \in \mathcal{M} \wedge O(\langle q_x, q_y \rangle) \geq \delta_O\}, \tag{3.12}$$

where δ_O defines a threshold on the minimum overlap – for higher values of δ_O, there is less chance of topical drift between queries.

3.1.3.2 Effectiveness Filter

The aim of fine-tuning T5 for the query reformulation task is to *improve* the retrieval effectiveness in response to an input query. Hence, when selecting training pairs, the retrieval performance should be taken into consideration, such that the target query q_y in the training dataset has a better quality than the source query q_x (in other words, the target query leads to a better retrieval performance than the source query). Therefore, we introduce an Effectiveness Filter, denoted as $\mathcal{W}_E$, to improve the quality of the initial pool. Given a query pair $\langle q_x, q_y \rangle$, we measure their relative effectiveness, with reference to relevance assessments L. Let $M(q)$ denote the effectiveness of ranking $R(q)$ for a particular effectiveness metric, such as reciprocal rank (RR) or discounted cumulative gain (DCG). Then, $\mathcal{W}_E$ filters the query pairs based on their relative effectiveness, as follows:

$$\mathcal{W}_E(\mathcal{M}) = \{\langle q_x, q_y \rangle \in \mathcal{M} \wedge (M(q_y) - M(q_x) > \delta_E\}, \tag{3.13}$$

where δ_E defines the required minimum positive change in M.

3.1.3.3 Stopwords Filter

Stopwords are function or grammatical words, e.g. “is”, or “and”, which have high term frequencies but contribute very little to enhancing the retrieval performance (Roy et al., 2019). However, many query pairs may consist of syntactical variations of the same query, such as stopwords that have changed order or have been substituted with other stopwords. In order to help the pretrained T5 model focus on generating terms that can identify relevant documents[1], we propose a Stopwords Filter – denoted as $\mathcal{W}_S$ – which refines the initial pool conditioned on the presence of the common words or stopwords. Given a pair of queries $\langle q_x, q_y \rangle$, to aid the T5 model producing non-generic words given any natural language query, the stopwords of the target query q_y are removed. Thus, in this case, the filter generates query pairs according to:

$$\mathcal{W}_S(\mathcal{M}) = \{\langle q_x, q_y - S_{stop} \rangle \in \mathcal{M}\}, \tag{3.14}$$

where, S_{stop} denotes a set of stopwords.

3.2 Research Questions

This paper focuses on addressing four research questions on our generative query rewriting models: Firstly, we investigate the effectiveness of the models under GenQR framework, namely the T5QR and FlanQR models. In particular, since the choice of training data is key to our fine-tuned T5-based reformulation models, we also investigate the best training settings among the weak supervision filters proposed in Section 3.1.3:

RQ3.1: What is the impact of the weak supervision training techniques for GenQR models?

Secondly, since we propose GenPRF, which makes use of contextualised input, we present our second research question:

RQ3.2: How does the additional pseudo-relevance contextual information affect the performance of GenPRF compared to GenQR, specifically when using T5PRF vs. T5QR and FlanPRF vs. FlanQR?

Thirdly, we compare the effectiveness of the studied generative reformulation models, including both T5-based and FLAN-based, with standard and recent query reformulation baselines by addressing the following research question:

RQ3.3: How do the studied generative reformulation models perform compared to the baseline query reformulation models?

Next, we investigate the effectiveness of the generative reformulation models within an advanced neural reranking pipeline. Hence, we propose the following fourth research question:

RQ3.4: Do queries reformulated using generative reformulation models result in further improvements when combined with a neural reranker?

Finally, we aim to determine the effect of the hyperparameters in our T5-based reformulation models.

RQ3.5: What is the impact of the number of top M passages in T5PRF, the number of paraphrases N to form the query reformulations as well as the relative weighting of reformulations obtained from RM3 and T5-based reformulation models?

[1] Indeed, in our experimental setup, stopwords are removed at retrieval time, so do not contribute to effectiveness.

Table 3.1: Summary of query pair training pools.

Method	Description	Pool Size	AvgLen(q_x)	AvgLen(q_y)
Initial pool	$Relevance(q_x) = Relevance(q_y)$	188,292	6.44	6.44
$\mathcal{W}_O$	$\delta_O = 5$	40,016	6.08	6.08
$\mathcal{W}_E$	$\delta_E = 0$	64,009	6.52	6.63
$\mathcal{W}_S$	$\langle q_x, q_y - S_{stop} \rangle$	188,292	6.44	3.46

3.3 Experimental Setup

In this section, we present the datasets used in our experiments in Section 3.3.1. Then, we describe the implementation details for GenQR and GenPRF models in Section 3.3.2 and present the configuration of the retrieval pipeline in Section 3.3.3. Finally, Section 3.3.4 provides details about the used baselines.

3.3.1 Datasets

As discussed in Section 3.1.3, one challenge when fine-tuning the T5 model for query reformulation is that there are no gold-standard labelled pairs of queries that indicate improved query formulations. To circumvent the lack of ground-truth data, we leverage the filters proposed in Section 3.1.3 to generate the required weak supervised query pairs to fine-tune the T5 model, instead of using human annotated query pairs. In our work, the training dataset is constructed based on the MSMARCO document ranking dataset as introduced in Section 2.5.1.

We follow the assumption introduced by Zerveas et al. (2019) that if a document is labelled as relevant for multiple queries, such queries are assumed to convey the same information need. Thus, we firstly identify 188,292 pairs of training queries that share the same labelled relevant document(s), then we further apply the filters introduced in Section 3.1.3 to reduce the noisy pairs, and increase the likelihood that they represent near-identical information needs.

For the Stopwords filter, we use a stopwords list of 733 words obtained from the Terrier IR platform. For the overlap filter, the top-K parameter is set to 10 as suggested by Mass et al. (2020) and the threshold value is configured as $\delta_O = 5$. For the effectiveness filter $\mathcal{W}_E$, we remove training pairs from the initial pool based on the difference in the discounted cumulative gain (DCG), using a minimum difference thresholds of $\delta_E = 0$. Statistics of the resulting pools after applying each filter are shown in Table 3.1. We do not select any more aggressive filter settings (cf. $\delta_O > 3$ or $\delta_E > 0$) as this results in pools smaller than 20-30k query pairs, which we found during our initial experiments to result in lowly performing query reformulation models.

We evaluate the retrieval effectiveness of our proposed T5QR framework on four standard test collections, namely: the TREC 2019 Deep Learning track (Craswell et al., 2021a) (i) document ranking and (ii) passage ranking tasks (both containing 43 queries), (iii) 250 queries from the TREC Robust 2004 track (Voorhees, 2004), and (iv) 149 description-only queries from the TREC Terabyte Track using GOV2 (Clarke et al., 2004). For the Robust 2004 test collection, we experiment using both the title-only and the description-only queries, to allow further comparisons with existing approaches from the literature.

3.3.2 Implementing GenQR and GenPRF

Fine-tuning the T5 Models: When implementing T5QR and T5PRF, we use the *t5-base* model with 220 million parameters (Raffel et al., 2020) obtained via the HuggingFace transformers library[2]. Following Raffel et al. (2020), fine-tuning is performed using a learning rate of 3×10^{-4} and a dropout rate of 10%. We train for 4 epochs, using a batch size of 6.

For the configuration of the T5PRF model using contextualised inputs, the size of the pseudo-relevance feedback set, K, is set to 10, following (Li et al., 2018, Zheng et al., 2020). A sliding window with a size of 128 and a stride of 64 tokens is used to split each feedback document into passages, following (Su et al., 2019). We select $M = \{1,2,3\}$ passages for use by the FirstP, TopP or MaxP selection approaches (cf. Equations (3.8) – (3.10)) as context input for our T5PRF model - this ensures that the maximum input length of the T5 model is not exceeded.

To generate the prediction sequence, the greedy decoding method is used. We use beam search with a beam size of 100 to form a large enough set of candidate paraphrases for each input query. Then we rank all the candidate paraphrases according to their likelihood, and select the N most likely paraphrases to form the reformulated query. In our experiments, we use $N = 5$, but return to the selection of N in Section 3.4.5.

Prompting the FLAN-T5 Models: When implementing FlanQR and FlanPRF models, we employ the *flan-t5-xxl* model with 3B parameters (Wei et al., 2021).[3] The configuration for extracting the pseudo-relevance feedback context information for FlanPRF is the same as the T5PRF. For an input query, we design task-aware prompts for FlanQR as follows, `FlanQR`(‘`Improve the search effectiveness by suggesting expansion terms for the query:` input query’). We also provide various examined prompts in Appendix A. The prompts for FlanPRF conditioned on the input query as well as its contextual information, `context(R)` is as follows, `FlanPRF`(‘`Improve the search effectivene-` `-ss by suggesting expansion terms for the query:` input query, `based on the given context information:` context(R)’). The selected prompts were identified among 20 candidate prompts based on their validation performance on the MSMARCO TREC 2019 passage ranking query set.

3.3.3 Retrieval Pipeline Setting

For ranking, we use a Porter stemmed index with stopwords removed. We apply a two-stage ranking pipeline, where the documents are first ranked by a tuned BM25 retrieval model using

[2] https://github.com/huggingface/transformers [3] The models’ sizes are selected based on our available computational capacity. Since T5QR and T5PRF require training, we were unable to use as large of a model as we were for the inference-only FlanQR and FlanPRF.

grid search. In the second stage, reranking is performed by the monoT5 model (cf. Section 2.2.2).

To deploy the monoT5 neural reranker, a sliding window is used to break up long documents into passages of 128 tokens with a stride of 64 between passages. We use MaxPassage (cf. Section 2.1.2) to obtain the final score of a document.

For the T5QR and T5PRF generated queries, we combine the generated query reformulation with the original query to form a new query for input in the first-stage retrieval, i.e. $q^{\dagger} = k_{RM3} \cdot q^{r}_{RM3} + k_{T5} \cdot q^{r}$, where k_{RM3} and k_{T5} are parameters that control the importance of reformulations query generated by RM3 or by the T5-based query reformulation model (T5QR or T5PRF). We use $k_{RM3} = 1$ and $k_{T5} = 0.5$ as a default setting, but investigate the impact of these two parameters in addressing RQ3.5.

For the FlanQR and FlanPRF generated queries, we append the generated query reformulations to the original query.[4] In particular, the importance of the generated query reformulations compared to the original query is controlled by a parameter β. We use $\beta = 0.2$ as the default setting in this work. All the hyperparameters are selected based on the validation performance on the TREC 2019 passage ranking query set.

3.3.4 Baselines

In order to evaluate the effectiveness of our proposed generative query reformulation models, we compare them to six families of query reformulation baselines, namely:

- **Initial query:** The original query without any modification.

- **Sparse Query Expansion:** Three traditional query expansion models, introduced in Section 2.4.1, are used as baselines, namely: (i) BM25+RM3, (ii) BM25+Bo1 and (iii) BM25+KL. We use BM25+RM3 as our main PRF baseline, due to its suitability as a baseline for neural methods (Lin, 2019).

- **Neural Query Reformulation:** (i) The Transformer model was employed by Zerveas et al. (2019) for the query reformulation task. We implement the transformer model using the OpenNMT platform (Klein et al., 2017); (ii) Sequence-to-Sequence Model with Attention. This model consists of an RNN-based encoder and decoder with an attention mechanism. The model is also implemented using the OpenNMT platform; (iii) GPT2 is a pretrained transformer-based language model. We fine-tune the GPT2 model for the query reformulation task as a baseline model.

- **Neural Document Expansion:** DocT5query is a document expansion model (cf. Section 2.2.3), which uses a T5 model fine-tuned to predict related queries for a given document.

[4] We tested versions for expansion terms both with and without RM3 terms for both T5 and FlanT5. T5 was more effective on our validation data with interpolated RM3 terms, while FlanT5 was more effective without. Therefore, we report these results.

Table 3.2: RQ3.1 - Part 1: Comparison between weak supervision filters on TREC 2019 document and passage ranking query sets. Superscripts a/b denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model. The highest value in each column is boldfaced.

QR Models	**Training/epoch**	**TREC 2019 Document**				**TREC 2019 Passage**			
		MAP	MRR	R@1k	nDCG@10	MAP	MRR	R@1k	nDCG@10
Initial query (a)	-	.341	.889	.701	.557	.311	.694	.752	.517
BM25+RM3 (b)	-	.397	.858	.760	.559	.342	.655	.796	.548
InitialPool	168 min	$.397^a$	.849	.757	.565	$.347^a$	.663	$.796^a$	.554
W_S	163 min	$.387^a$	.833	.745	.579	.342	.654	.777	.532
$\mathcal{W}_O$	35 min	$.398^a$	.860	$.756^a$	.581	.343	.644	$.797^a$	.543
$\mathcal{W}_E$	55 min	$\mathbf{.400}^a$	.856	$.761^a$	.581	$.348^a$	.657	$.808^a$	.547
$\mathcal{W}_{(O+S)}$	34 min	$.397^a$	.875	$.756^a$	.580	.344	.634	.793	.542
$\mathcal{W}_{(E+S)}$	54 min	$.398^a$	**.914**	$.753^a$	**.600**	$.351^a$	$.645^a$	.788	.535
FlanQR	-	$.384^a$	.790	$\mathbf{.763}^a$	.537	$\mathbf{.382}^a$	**.707**	$\mathbf{.845}^a$	**.556**

These are then appended to the original document during indexing time. We obtain the results of DocT5query for the MSMARCO document corpus from (Ma et al., 2022).

- **CEQE Models:** CEQE (cf. Section 2.4.2) is a BERT-embedding based query expansion model. In particular, 70-100 expansion terms selected in the contextualised embedding space are added to the original query to form a new query. We compare with three variants of CEQE, namely CEQE-Max, CEQE-Centroid and CEQE-mul models. In our implementation, we apply CEQE query expansion models upon the documents retrieved by BM25 and apply the pipeline BM25 + RM3 + BM25 rather than the Dirichlet LM + RM3 +BM25 pipeline which is used by the original CEQE implementation (Naseri et al., 2021).
- **Neural Query Expansion Models:** (i) NPRF (cf. Section 2.4.2) is a neural pseudo-relevance feedback model, which operates as a reranker, that it does not generate a refined textual query that is re-applied to the inverted index, but instead uses the pseudo-relevance feedback to re-ranked the initial candidate set of documents. The best variant – NPRF$_{ds}$-DRMM – is included as a baseline; (ii) BERT-QE (cf. Section 2.4.2) is also a neural PRF model that builds on NPRF, but use BERT (Large) to refine the PRF information. We compare with the reported best variant, BERT-QE-LLL.

3.3.5 Evaluation Metrics

Following standard practice in the TREC Deep Learning track (Craswell et al., 2021a) that introduced in Section 2.5.3, we measure the effectiveness of the reformulated queries through their ranking performance in terms of Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR)[5], as well as Recall and normalised discounted cumulative gain calculated to rank depth

[5] Although there has recently been some discussion about the choice of the MAP and MRR metrics to report in the IR community, no consensus has yet been reached. Thus, to compare with the existing works, we follow the widely used metrics in our work.

Table 3.3: RQ3.1- Part 2: Comparison between weak supervision filters on Robust04 and GOV2 query sets. Superscripts a/b denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model. The highest value in each column is boldfaced.

QR Models	**Training/epoch**	**Robust04 (T)**			**Robust04 (D)**			**GOV2**		
		MAP	R@1k	nDCG@10	MAP	R@1k	nDCG@10	MAP	R@1k	nDCG@10
Initial query (a)	-	.256	.705	.439	.233	.670	.411	.268	.643	.468
BM25+RM3 (b)	-	.284	.752	.437	.266	.709	.427	.291	.657	.452
$InitialPool$	168 min	.286^{a}	.754	.443	.271^{ab}	.721^{ab}	.434^{b}	.293^{ab}	.662^{b}	.461
W_S	163 min	**.290**^{ab}	**.760**	.444	**.279**^{ab}	.736^{ab}	.439	.301^{ab}	.680^{ab}	.474
$\mathcal{W}_O$	35 min	.285^{a}	.753	.442	.268^{ab}	.711^{ab}	.428	.294^{ab}	.663^{b}	.463
$\mathcal{W}_E$	55 min	.286^{a}	753	.442	.269^{ab}	.714^{ab}	.429	.292^{ab}	.658^{b}	.460
$\mathcal{W}_{(O+S)}$	34 min	.286^{ab}	.744	. 442	.270^{ab}	.718^{ab}	.429	.302^{ab}	.675^{ab}	.472^{b}
$\mathcal{W}_{(E+S)}$	54 min	.287^{ab}	.753	.442	.278^{ab}	.734^{ab}	.439	.302^{ab}	.676^{ab}	.474^{b}
$FlanQR$	-	.270^{ab}	.756^{a}	**.461**	.278^{a}	**.760**^{ab}	**.471**^{ab}	**.303**^{ab}	**.680**^{ab}	**.510**^{ab}

10. We use the paired t-test ($p < 0.05$) for significance testing and apply the Holm-Bonferroni multiple testing correction.

3.4 Results and Discussions

We now address our five research questions in turn: the impact of the weak supervision filters on training data quality for GenQR models (Section 3.4.1); the usefulness of pseudo-relevance feedback as contextualised input, as per our proposed GenPRF model (Section 3.4.2). Then, we make comparisons with the baselines (Section 3.4.3) and consider the role of a neural reranker (Section 3.4.4). Next, we investigate the hyperparameter of GenQR and GenPRF models in Section 3.4.5. We also provide qualitative analysis of all the GenQR and GenPRF models in Section 3.4.6. Finally, we position our proposed methods within the prior approaches and discuss the difference between GenQR and GenPRF models in Section 3.4.7.

3.4.1 RQ3.1: Impact of the Training Data Quality

In order to better fine-tune the T5QR model, in Section 3.1.3, we proposed to generate refined training pools using different weak supervision filters. Table 3.2 compares the average training time needed per-epoch and the effectiveness of T5QR trained with three different weak supervision filters, namely the Stopwords Filter, the Overlap Filter and the Effectiveness Filter (top half), as well as using combinations of these filters (bottom half). In addition, we also report the retrieval effectiveness for the FlanQR model in Table 3.2 & Table 3.3, which does not require fine-tuning. We report results on all four test collections: TREC 2019 for document ranking and passage ranking tasks (in Table 3.2), as well as GOV2 and Robust04 (in Table 3.3).

From Table 3.2 & Table 3.3, we observe that both T5QR trained with the initial pool and the smaller pools result in significant improvements over the initial query on all datasets and

Table 3.4: RQ3.2: Effect of the PRF contextualised input on TREC 2019 document and passage ranking query sets. Superscripts a/b/c/d denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model(s). The highest value in each column is boldfaced.

QR Models	**TREC 2019 Document**				**TREC 2019 Passage**			
	MAP	MRR	R@1k	nDCG@10	MAP	MRR	R@1k	nDCG@10
Initial query (a)	.341	.889	.701	.557	.311	.694	.752	.517
BM25+RM3 (b)	.397	.858	.760	.579	.342	.655	.796	.548
$T5QR$ (d)	.398^{a}	**.914**	.753	.600	.351	.645	.788	.535
$T5PRF_{FirstP}$	**.411**^{a}	.868	.770	.596	-	-	-	-
$T5PRF_{TopP}$	.411^{a}	.868	.776^{a}	.602^{a}	.353^{a}	.676	.815^{a}	.554
$T5PRF_{MaxP}$	.411^{a}	.868	.776^{a}	.602^{a}	-	-	-	-
$FlanQR$ (e)	.384^{a}	.790	.763^{a}	.537	.382^{a}	.707	.845^{a}	.556
$FlanPRF_{FirstP}$	.373	.816	.841	**.605**	-	-	-	-
$FlanPRF_{TopP}$	.373	.816	.841	**.605**	**.404**^{ab}	**.809**^{b}	**.866**^{ab}	**.628**^{ab}
$FlanPRF_{MaxP}$	.361	.780	**.844**	.578	-	-	-	-

Table 3.5: RQ3.2: Effect of the PRF contextualised input on Robust04 and GOV2 query sets. Superscripts a/b/c/d denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model(s). The highest value in each column is boldfaced.

QR Models	**Robust04 (T)**			**Robust04 (D)**			**GOV2**		
	MAP	R@1k	nDCG@10	MAP	R@1k	nDCG@10	MAP	R@1k	nDCG@10
Initial query (a)	.256	.705	.439	.233	.670	.411	.268	.643	.468
BM25+RM3 (b)	.284	.752	.437	.266	.709	.427	.291	.656	.452
$T5QR$ (d)	.288^{a}	.756^{a}	.443	.277^{ab}	.734^{ab}	.439^{ab}	.302^{ab}	.676^{ab}	.474^{b}
$T5PRF_{FirstP}$	**.297**^{ab}	.757^{a}	**.463**^{bd}	.282^{abd}	.740^{abd}	.441^{abd}	.302^{ab}	.676^{ab}	.475
$T5PRF_{TopP}$	.296^{ab}	**.759**^{a}	.461^{bd}	**.285**^{abd}	.743^{abd}	.447^{abd}	**.303**^{ab}	**.685**^{ab}	.479^{b}
$T5PRF_{MaxP}$	.296^{ab}	.759^{a}	.461^{bd}	**.285**^{abd}	.743^{abd}	.447^{abd}	**.303**^{ab}	**.685**^{ab}	.479^{b}
$FlanQR$ (e)	.270^{ab}	.756^{ab}	.461^{ab}	.278^{a}	**.760**^{ab}	**.471**^{ab}	**.303**^{ab}	.680^{ab}	**.510**^{ab}
$FlanPRF_{FirstP}$	.260	.699	.434	.225	.656	.433^{b}	.258	.597	.490^{b}
$FlanPRF_{TopP}$	.260	.699	.434	.224	.656	.406	.258	.597	.490^{b}
$FlanPRF_{MaxP}$	.243	.683	.420	.225	.656	.433^{b}	.257	.597	.496^{b}

significantly outperforms BM25+RM3 for a few settings and measures, particularly on Robust04 (D) and GOV2. In addition, we also note that applying the Overlap filter and the Effectiveness filter only takes 33% and 20% of the training time needed for the initial pool, respectively. Among the filters, the performances achieved are similar, and hence we conclude that there is no need to use the initial pool as the training time is less with more aggressive filters. Moreover, we observe that when combined with the stopwords filter, these combined filters demonstrate marked performance enhancement over the single filters. Thus, to avoid spending time training the T5 model to generate useless stopwords that do not affect the BM25 retrieval process, we take forward the higher recall combined filter, i.e. $\mathcal{W}_{(E+S)}$, when addressing RQ3.2.

Furthermore, we also report the performance of FlanQR, which does not require fine-tuning. We observe that FlanQR leads to significant improvements over the initial query, which indicates the usefulness of the prompting-generated query reformulations.

Table 3.6: RQ3.2: Comparison with the baseline query expansion approaches. Superscripts a...j denote significant improvements over the indicated baseline model(s).

Models	**TREC 2019 Doc.**			**Robust04 (T)**		**Robust04 (D)**		**GOV2**	
	MAP	nDCG@10	nDCG@20	MAP	nDCG@20	MAP	nDCG@20	MAP	nDCG@20
BM25+Bo1 (a)	.384	.567	.538	.287	.433	.271	.418	.295	.455
BM25+KL (b)	.391	.570	.558	.290	.435	.271	.419	.291	.458
BM25+RM3 (c)	.397	.579	.558	.284	.427	.266	.408	.291	..452
Seq2seq$_{attention}$ (d)	.255	.414	.403	.199	.343	.204	.343	.216	.377
Transformer (e)	.238	.421	.423	.214	.366	.207	.347	.216	.377
GPT2-QR (f)	.327	.518	.490	.237	.409	.237	.396	.265	.407
DocT5query (g)	-	.597	-	-	-	-	-	-	-
CEQE-Max (h)	**.419**	.554	.541	.300	.425	**.290**	.433	.283	.455
CEQE-Centroid (i)	.417	.550	.541	.299	.423	.289	.431	.286	.453
CEQE-Mul (j)	.410	.535	.541	.292	.415	.285	.424	.289	.448
NPRF	-	-	-	.290	.450	.280	**.456**	-	-
BERT-QE (LLL)	-	-	-	**.386**	**.553**	-	-	.268	**.604**
T5QR	.398^{def}	.600^{def}	.583^{def}	.288^{cdef}	.432^{cdef}	.278^{bcdef}	.420^{cef}	.302^{cdef}	.460^{cde}
T5PRF	.411^{def}	**.605**^{def}	**.583**^{def}	.296^{cdef}	.449^{cdef}	.285^{abcdef}	.428^{cdef}	**.303**^{cdef}	.465de
FlanQR	.384de	.537^{def}	.520^{def}	.270^{def}	.436^{def}	.278^{def}	.446^{def}	**.303**^{cdef}	.496^{def}
FlanPRF	.373de	**.605**^{def}	.577^{def}	.260^{def}	.411^{def}	.224^{def}	.393^{def}	.262de	.464de

Overall, in response to RQ3.1, we find that both the T5QR and FlanQR query reformulation models can significantly improve over the initial query and, in some cases, BM25+RM3. For T5QR, using filters can lead to faster training without loss of effectiveness.

3.4.2 RQ3.2: Effect of PRF Contextualised Input

We now investigate the effect of the additional PRF contextual information for T5PRF and FlanPRF models. In particular, we investigate three selection mechanisms introduced in Section 3.1.2, namely MaxP, FirstP and TopP. For the number of context passages we use $M = 1$ – we return to this choice in Section 3.4.5. Table 3.4 & Table 3.5 reports the results of our experiments for RQ3.2. We again evaluate using all four test collections.

First, we analyse the performance of T5PRF models. Examining both Table 3.4 & Table 3.5, we find that in each group of models, the T5PRF models exhibit some marked improvements over T5QR for all metrics on all query sets, indicating the effectiveness of the contextualised input. When comparing to the BM25+RM3 model, marked improvements in terms of MAP, Recall and nDCG@10 are also observed. In particular, the T5PRF models significantly outperform the BM25+RM3 model in terms of MAP and nDCG@10 on the Robust04 (T), Robust04 (D), and GOV2 query sets, as well as in terms of Recall on Robust (D) and GOV2. This observation indicates that query reformulations generated by the T5PRF models are capable of retrieving relevant documents that are not identified by the RM3 reformulated queries.

Next we analyse the contextual prompting method, FlanPRF. We observe that FlanPRF exhibits higher performance over FlanQR on all the reported metrics for TREC 2019 passage and document query sets, except on MAP for document queries. This indicates the superiority of the additional contextual information for generating more useful expansion terms on these queries. However, on the Robust title & description queries as well as the GOV2 queries, FlanPRF gives lower performance compared to FlanQR. We postulate that the FLAN-based models need more carefully crafted domain-related prompts, perhaps as some of the instruction data for fine-tuning the FlanT5 model comes from the MSMARCO Q&A training datasets (Wei et al., 2021).

Finally, among the various contextual information selection mechanisms, we find that TopP and FirstP exhibit higher performance than the MaxP method. This observation indicates that the beginning sentences are able to capture the meaning of the whole document. In addition, no differences between the TopP and FirstP methods are observed for the T5PRF and FlanPRF models. Therefore, we take forward the TopP method, where the top-scored chunks among all the feedback information are selected as the contextual input, for addressing RQ3.3 and RQ3.4.

Overall, for RQ3.2, we observe that PRF information, in the form of contextual passages, can bring further improvements over the plain T5QR model on all the five test query sets. However, FlanPRF only benefits from the additional pseudo-relevant information for MSMARCO document and passage query sets and damages the retrieval effectiveness on Robust and GOV2 queries. Finally, the performance of GenPRF models varies according to the context selection mechanism.

3.4.3 RQ3.3: Comparison with Baselines

In this section, we examine the effectiveness of the GenQR and GenPRF models, in comparison to the baselines listed in Section 3.3.4, including traditional query expansion models, neural query reformulation baselines, neural document expansion baselines, and neural query expansion models from the literature. Due to space constraints, Table 3.6 compares the performances in terms of MAP and nDCG@20 on the three test collections where the Neural PRF and BERT-QE baselines have been evaluated in the literature: TREC 2019, GOV2, as well as the title-only (T) and description-only (D) queries of Robust04.[6] For these baselines, we omit performances not reported in the original papers.

On analysing Table 3.6, we first observe that the GenQR models significantly outperform the other generative neural query reformulation models, which have been trained on the same training input (namely GPT2-QR, Seq2seq$_{attention}$ and Transformer). This emphasises the usefulness of using T5 and FlanT5 models over other text-to-text approaches such as GPT2.

Next, we compare the GenPRF models with standard PRF query expansion baselines such as RM3, Bo1 and KL. We observe that on the Robust04 (T) query set, the T5PRF models exhibit significant improvements over the BM25+RM3 model for both metrics and significantly outperform the BM25+KL model in terms of MAP on Robust04 (D), due to the good ability of the T5 model in interpreting and reformulating the natural language queries posted in the description (D) queries.

Furthermore, in comparison to DocT5query on the TREC 2019 document test query set, we

[6] For Robust04, we report nDCG@20 to allow comparisons to be made to (Li et al., 2018, Naseri et al., 2021, Zheng et al., 2020).

Table 3.7: RQ3.4 Part1: Performances of T5QR and T5PRF using the monoT5 reranker on TREC 2019 document and passage ranking query sets. Notations as in Tables 3.2.

QR Models	**Document TREC 2019**				**Passage TREC 2019**			
	MAP	MRR	R@1k	nDCG@10	MAP	MRR	R@1k	nDCG@10
Initial query (a)	.388	.938	.701	.688	.480	.857	.752	.711
BM25+RM3 (b)	.405	.938	.760	.693	.489	.833	.796	.708
T5QR	.394	.938	.753^{a}	.690	.479	.831	.788^{a}	.696
T5PRF	.403	.926	.776^{a}	.687	.490	.826	.814^{a}	.706
FlanQR	**.413**^{a}	**.950**	**.763**^{a}	**.699**	.521^{a}	**.908**	.845^{a}	**.727**
FlanPRF	.406	.942	.762^{a}	.697	**.530**^{a}	.873	**.866**^{a}	.724

Table 3.8: RQ3.4 - Part2: Performances of T5QR and T5PRF using the monoT5 reranker on Robust04 and GOV2 query sets. Notations as in Tables 3.2.

QR Models	**Robust04 (T)**			**Robust04 (D)**			**GOV2**		
	MAP	R@1k	nDCG@10	MAP	R@1k	nDCG@10	MAP	R@1k	nDCG@10
Initial query (a)	.269	.708	**.481**	.280	.672	.517	.262	.643	.532
BM25+RM3 (b)	.267	.755	.475	.290	.712	.516	.263	.656	.517
T5QR	**.269**	.759^{a}	.474	.295	.736	.524	.269^{b}	.676^{ab}	.526
T5PRF	**.269**	**.759**^{a}	.474	.299^{ab}	.746^{ab}	.525	**.273**^{ab}	**.685**^{ab}	.535^{b}
FlanQR	.265	.756^{a}	.473	**.301**^{ab}	**.760**^{ab}	**.536**^{ab}	.271^{a}	.680^{ab}	**.543**^{ab}
FlanPRF	.254	.699	.466	.275	.656	.517	.241	.597	.524

observe that both T5PRF and GenPRF models exhibit higher performances than the DocT5query model in terms of nDCG@10. This observation demonstrates the effectiveness of our generative reformulation models, which do not require applying the GPU-intensive application of a T5 model to each document in the collection, while DocT5query does.

When comparing with the CEQE models, we find that the T5PRF models exhibit similar performances in terms of MAP but much higher performances in terms of nDCG@20 for TREC 2019 Document ranking query set. For the Robust title and description query sets, T5PRF models show similar performances to the CEQE variants on both metrics. Finally, T5PRF models exhibit slightly higher performance than the CEQE models on both metrics for the GOV2 queries. In addition, another strength of T5PRF models compared to the CEQE variants is that T5PRF models only add ∼10 expansion terms rather than the 70-100 expansion terms added by CEQE.

Finally, we compare with the results reported for Neural PRF, BERT-QE. It is clear that T5PRF performs very similarly to Neural PRF (e.g. for MAP 0.296 vs. 0.290 on title-only queries and 0.285 vs. 0.280 on description-only queries). However, as can be observed in Table 3.6, BERT-QE, which deploys three stages of BERT-Large (in comparison to our use of the comparably simpler T5-base model), exhibits a higher performance than our (simpler) T5PRF approach, which does not employ any neural reranking. Overall, we conclude for RQ3.3 that T5PRF offers a promising approach for neural query expansion, which significantly outperforms existing statistical query expansion approaches.

3.4.4 RQ3.4: Integration with Neural Rerankers

We now investigate the effectiveness of GenQR and GenPRF models when combined with a monoT5 neural reranker, in Table 3.7 & Table 3.8. We integrate the reformulated queries with the monoT5 neural reranker as described in Section 3.3.3.

On analysing both Table 3.7 & Table 3.8, we observe that when combined with the monoT5 reranker, the T5PRF models exhibit higher performances than the T5QR model, which aligns with the conclusions of RQ3.2. Moreover, similar to the findings of RQ3.2, FlanPRF leads to higher effectiveness than FlanQR on MSMARCO document and passage queries. We also observe that all the generative models, except FlanPRF, significantly outperform the BM25 with monoT5 reranking across all datasets in terms of Recall. This suggests that the reformulated queries generated by T5PRF are more effective in retrieving relevant documents compared to the original query or the query reformulated using BM25. For FlanPRF, we find that it excels at the MSMARCO queries but fails to produce a better reformulation for Robust and GOV2 queries. Moreover, we find that T5PRF models show considerable improvements over both baselines in terms of MAP and Recall for the Robust04 dataset, which uses only descriptions, and the GOV2 query sets. We note that higher performance on Robust has been reported using monoT5 by Nogueira et al. (2020), but with an experimental setup that is not directly comparable with ours[7]. In our experiments, we use a more realistic setting where the user is expected to only enter either keyword-based queries or natural-language queries, not both. Overall, in response to RQ3.4, we conclude that our generative models can further enhance effectiveness when combined with neural rerankers.

3.4.5 RQ3.5: Hyperparameter study

We address RQ3.5 by analysing the importance of the hyperparameters of T5QR and T5PRF. We first consider the impact of the number of selected passages, M, used as the contextualised input to T5PRF. Recall that the maximum input to the pretrained T5 model, X, is limited to 512 tokens; for this reason, and to ensure sufficient space for the initial query and the prompt tokens, M cannot exceed 3 passages, given that we use passages of 128 tokens (cf. Section 3.3.2).

Figure 3.3 presents the MAP and nDCG@10 scores of the T5PRF models using different M values on the TREC2019 document query set. We observe that while nDCG@10 is improved with more passages (i.e. as M increases), MAP tends to be degraded. We postulate that when more passages are selected as an input, it is more likely that the T5 model will generate a few off-topic terms (i.e. topic drift), which will have low weights but can negatively affect MAP; On the other hand, with more passages, the very important terms for the most highly relevant passages are more easily identified and emphasised in the resulting query reformulation, resulting in improved nDCG@10.

Next we investigate the impact of the k_{RM3} and k_{T5} parameters introduced in Section 3.3.3. Figure 3.4 shows a heatmap depicting the performance of a T5PRF model using different values of

[7] As (Nogueira et al., 2020) makes use of *both* the title (T) and description (D) versions of the query, which is not a realistic setting from a user's perspective.

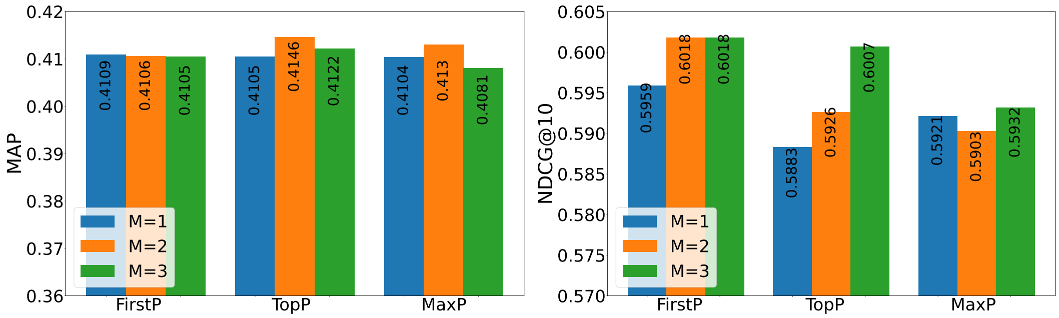


Figure 3.3: Impact of the number of passages M in T5PRF. Different coloured bars are shown for the three different contextual passage selectors FirstP, TopP and MaxP.

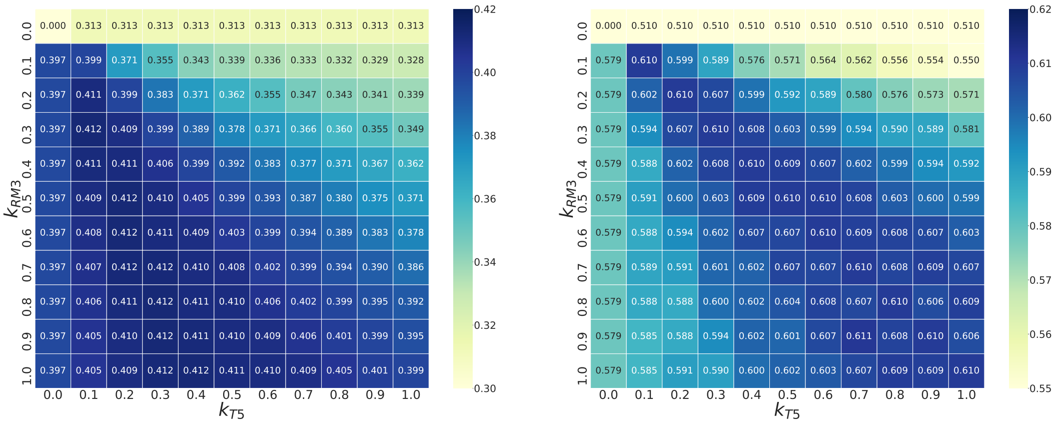


Figure 3.4: Impact of the mixing parameters k_{RM3} and k_{T5}.

k_{T5} (x-axis) and k_{RM3}(y-axis) on the TREC 2019 Document task, for (a) MAP and (b) nDCG@10. Interestingly, we note differences in the overall patterns between MAP and nDCG@10. For both metrics, the highest values are generally on or near to the diagonal; while effectiveness drops off for low values of k_{T5}. This is more marked for nDCG@10 than MAP. Therefore, we conclude that when the T5-generated query reformulations are combined with RM3, there is more impact to the top of the ranking, as quantified by the larger improvements in nDCG@10 compared to MAP.

Finally, we examine the impact of the number of paraphrases N selected from a T5-based model to construct the reformulated query. We take a trained model of T5PRF using the $W_{(E+S)}$ filter as an example. Figure 3.5 illustrates the impact on the MAP and nDCG@10 scores when varying $N = [1, 20]$. We observe that, in general, the higher the number of paraphrases to be included in the query reformulation, the higher the values that MAP or nDCG@10 are likely to achieve, although with a degree of variance. In particular, all tested N values markedly exceed the corresponding performance of BM25+RM3. Note that while we only chose $N = 5$ in reporting the experiments in this paper in order to facilitate faster retrieval, larger N values usually lead to a higher effectiveness.

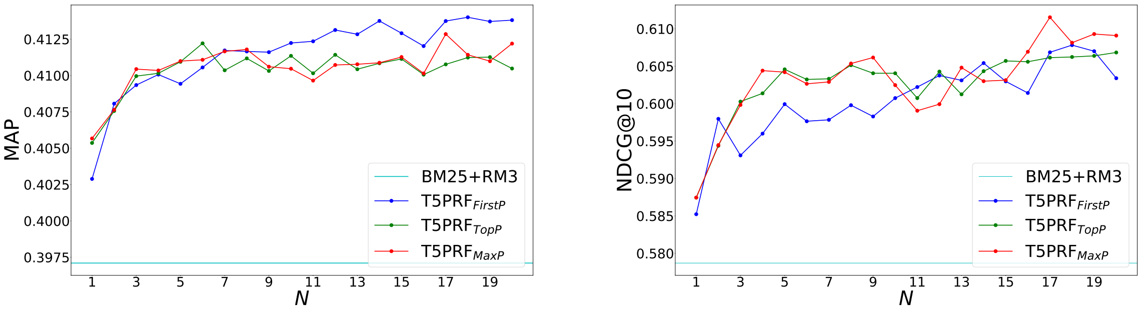


Figure 3.5: Impact of the number of paraphrases *N* obtained from T5 to form a query reformulation. Plots are shown for MAP (left) and nDCG@10 (right).

3.4.6 Case Study

Table 3.9 provides (stemmed) example reformulations for a query from the TREC 2019 test query set using the RM3, the GenQR and GenPRF query reformulation models..

In particular, all the GenQR and GenPRF models can generate new useful terms that are not identified by the RM3 model. For T5-based models, we observe that both T5QR and T5PRF models tend to generate more conservative terms, which present less risk for topical drift. In addition, we observe that, in Table 3.9, T5PRF focuses on generating expansion terms that are very closely related to the definition of *visceral*, such as ‘viscera’, ‘body’. On the other hand, RM3 identifies terms more widely related to the body, such as ‘cardiac’, and ‘fat’, but which are less likely to identify more relevant documents for this query.

For FLAN-based models, under the GenQR framework, we find that FlanQR can generate query reformulations consisting of natural language definitions of the *visceral* from various aspects, for instance, the functionality of visceral, the further explanation of *viable organ* etc. However, with the additional contextual information, some of the reformulated sentences tend to focus on *abdominal*, which is provided in the contextual input, instead of the query term *visceral*. Therefore, we find that FlanPRF can easily be drifted away and might struggle to locate useful information from the potentially relevant information.

3.4.7 Discussion

In the previous sections, we investigated the performance of all four proposed generative query reformulation methods compared to various query reformulation baselines. In this section, we first position our proposed models within the prior approaches and discuss how the prompting-based models, FlanQR and Flan-PRF, perform compared to the fine-tuning-based models, T5QR and T5PRF.

Compare to Prior Approaches: Several techniques have been proposed to employ the PLM knowledge for query reformulation for various tasks. For instance, for the OpenQA task, Mao et al. (2021) used a pretrained language model as a generator to produce background context information for an input question – this generated context together, with the input question. However, this framework is not applicable for addressing an adhoc search task due to the lack of a training dataset, e.g. query and ground truth answer training pairs, while the produced augmented queries are long in length which can result in high query latency during retrieval. Similarly, Mass et al. (2020) fine-tuned a GPT-2 model using FAQ (Frequently Asked Questions) pairs to generate the predicted questions given an input query. In addition, T5 or GPT2 have been employed as neural transfer reformulation models to reformulate the query based on the recent querying history in conversational search (Lin et al., 2020b,c, Yu et al., 2020). Furthermore, Lin et al. (2020b) compared pretrained models using an encoder-decoder architecture (T5 model) and a GPT2 (decoder only) model for conversational question reformulation, and concluded that T5 is more effective. In contrast, our work focuses on proposing a generative method for query reformulation and concentrates on the well-optimised sparse adhoc retrieval task. In particular, for fine-tuning a trained language model, we leverage weak supervision methods to generate the necessary training query pairs, which are then used to fine-tune the existing pretrained T5 model for the query reformulation task.

Moreover, while prior work has used smaller open-source models, there has been a recent shift to very large, proprietary, closed-source text generation models, such as OpenAI's *text-davinci-003* with 175 billion parameters. Concurrently with this work, Wang et al. (2023a) proposed the query2doc method, which expands the query with generated pseudo-documents from *text-davinci-003*. In addition, Mackie et al. (2023) also prompted *text-davinci-003* for query expansion using various prompting techniques. Unlike these works, we provide a comparison between the effectiveness of fine-tuning and prompting methods for a particular task. Further, we explore multiple frameworks for query reformulation, namely GenQR and GenPRF, and investigate how to more effectively employ the learned knowledge from the large neural models. In addition, it is well-established that large language models (LLMs) with parameters exceeding 100 billion have superior performance on various tasks (Brown et al., 2020, Chowdhery et al., 2022, Wei et al., 2022a). However, using these large-scale models is often prohibitively expensive. Therefore, our work explores the ability to use small models with parameters of less than 10 billion for which inference can be performed on consumer GPUs.

Architecture: GenQR methods do not rely on a set of initial retrieved results.[8] FlanQR has the simplest model structure as it also does not require fine-tuning of the model's parameters. However, FlanQR is sensitive to the input prompts to generate good query reformulations. In contrast, T5QR involves injecting task-related knowledge into the model's parameter during fine-tuning, but once the model is trained, there is no need for prompt crafting during inference.On the other hand, GenPRF methods depend upon an initial round of retrieval to provide pseudo-relevant contextual information. T5PRF is fine-tuned in a weakly supervised way and can better extract

[8] We note that although we found T5QR to be most effective when interpolated with RM3 expansion terms, the initial retrieval process for RM3 expansion can be conducted in parallel with T5QR inference, since it does not depend on the first-stage results.

useful information from the PRF contextual information. However, the only input information source for FlanPRF is prompt. As the prompt can be long, it may contain too much information for a FlanPRF model to discern which pieces of text can make a good query reformulation.

Challenges: In fine-tuning-based methods, the primary challenge lies in constructing high-quality training data for a specific task to adapt the model's capacity to address the target task. Evidence for this can be found in the results presented in Table 3.2 &Table 3.3. On the other hand, prompting methods do not involve changing the model's parameters and only rely on the model's existing knowledge and understanding to produce desired outputs. The key challenge for prompting methods is to identify task-related prompts that effectively unlock the large language model's learned knowledge. Moreover, based on the qualitative study conducted in Section 3.4.6, we find that the zero-shot prompting method can be challenging for the FlanPRF method in terms of identifying useful information and filtering out distracting details from the contextual information. Moreover, one might resort to few-shot prompting by providing a prompt with a few examples for generative models to learn from. However, few-shot prompting also encounters similar challenges with T5-based methods, such as constructing high-quality examples and limitations in input length. Overall, based on the effectiveness results for FlanQR and FlanPRF models observed in Tables 3.4-3.8, it is still promising to explore the effective way of designing query reformulation task-related prompts for the LLMs. For instance, instead of designing one prompt template and deploying it across various datasets, different prompts may be needed for different datasets.

3.5 Conclusions

In the proposed thesis statement in Section 1.1, we posited that we can use pseudo-relevance feedback information by a Sequence-to-Sequence neural model to generate more effective query reformulations for sparse retrieval. Therefore, to address this hypothesis, in this chapter, we investigated neural query reformulation methods built upon the generative neural models with Sequence-to-Sequence architecture, such as T5 and FLAN-T5 models. In particular, we proposed two possible generative query reformulation frameworks, GenQR and GenPRF. Models under the GenQR framework directly take a query as input, while models under the GenPRF framework also incorporate contextual information extracted from the pseudo-relevant feedback documents. Moreover, under each framework, we investigated both fine-tuning and direct prompting methods to leverage the learned knowledge of T5 and FLAN-T5, respectively. Extensive experiments showed that the GenQR and GenPRF models can significantly enhance effectiveness on four standard TREC test collections (when using either keyword or natural-language queries) and that significant improvements can also be observed when combined with additional neural rerankers compared to standard PRF techniques such as RM3.

In conclusion, the main findings of this chapter can be summarised as follows:

- We find that pseudo-relevance feedback information, in the form of contextual input, can bring further improvements over the plain GenQR model on all five test query sets. However, the performance of the GenPRF models, i.e. T5PRF and FlanPRF, varies depending on the context selection mechanism.

- GenPRF models, in particular, the T5PRF model offers a promising approach for neural query expansion, which significantly outperforms the existing statistical query expansion approaches.
- When interrogating the neural rerankers, we find that our generative models can further enhance effectiveness when combined with neural rerankers.

Overall, we found that the pseudo-relevance feedback information can be used by the Sequence-to-Sequence neural model in a generative way to refine the query formulation for more effective sparse retrieval. Specifically, our generative query reformulation models can produce queries that are more precise than RM3 (in terms of nDCG@10) while also enhancing Recall. Moreover, compared to competing techniques such as docT5query, our studied models can be applied at querying time without the need to apply expensive neural models to all documents at indexing time.

However, there are also some limitations to the proposed generative query reformulation methods. Firstly, any prompting-based generative query reformulation methods depends heavily on the input prompt. Secondly, the primary focus of these methods is on effective sparse retrieval. As we detailed in Chapter 2, there are several paradigms for more advanced neural information retrieval, namely the retrieve-then-rerank paradigm (cf. Section 2.2.2) and dense retrieval (cf. Section 2.3). In addition, this chapter mainly explored the capability of the Sequence-to-Sequence PLMs for more effective query reformulation, the generated query reformulations are still in text format. Moreover, as detailed in Section 2.2.2 and Section 2.3, the encoder-based PLMs, in particular, the BERT model, have shown strong capabilities for generating contextualised embeddings and employed as effective neural rerankers as well as the dense retrieval models. However, the effectiveness of the pseudo-relevance feedback mechanism for the more advanced dense retrieval is still limited study. Therefore, in the following chapters, from Chapter 4 to Chapter 7, we will investigate the possibility of pseudo-relevance feedback techniques for more effective dense retrieval.

Table 3.9: Example reformulations of ‘define visceral’ using RM3, T5QR & T5PRF as well as FlanQR & FlanPRF approaches. To aid the reading, we highlight the prompt words in red and the original query in blue colour.

BM25+RM3 Input Query: define visceral
BM25+RM3 Reformulated Query:
central∧0.0253 defin∧0.3167 cell∧0.0503 fat∧0.0371 obes∧0.0384 muscl∧0.077 cardiac∧0.0201 viscer∧0.3833 tissu∧0.0249 skelet∧0.0261

T5QR Input Query: "refine": define visceral
T5QR Reformulated Query:
viscer∧0.1041 definit∧0.1041 viscera∧0.0327 defin∧0.0327 definit∧0.0253 viscer∧0.0253 viscer∧0.0135 fluid∧0.0135 viscer∧0.0098 structur∧0.0098

T5PRF Contextualised Input Query:
"refine": define visceral. "context": viscera on 5 Apr.2018. These example sentences are selected automatically from various online news sources to reflect current usage of the word ‘visceral’. Views expressed in the examples do not represent the opinion of Merriam-Webster or its editors. Send us feedback. The Tie Between visceral and Biology. The “viscera” are the internal organs of the body-especially those located in the large cavity of the trunk (e.g., the heart, liver, and intestines). The word viscera comes from Latin, in which it has essentially the same meaning.Something “visceral” has to do with the viscera. In a more figurative sense, something “visceral” is felt “deep down”. Even in the early years of its use, “visceral” often referred to things emotional rather than physiological. For example, in 1640, an English bishop named Edward Reynolds wrote, “Love is of all other the inmost”.
T5PRF Reformulated Query:
viscera∧0.1596 viscer∧0.1556 viscera∧0.0592 bodi∧0.0592 viscer∧0.0379 fat∧0.0379 definit∧0.0243 viscer∧0.0243

FlanQR Input Query: Improve the search effectiveness by suggesting expansion terms for the query: define visceral
FlanQR Reformulated Query:
Viable organ means the possibility that it could be dead if not harvested and the normal, healthy human being is capable of dead Organs, substances in organ meat and whole part of digestion
Viscera (and viscerally) Able:-able; competent, eager to serve
Visceral or viscular can mean the inner layer, which includes any cell wall-containing tissue lining cells in organs
Visceral art refers to the expressive medium of artists, drawing on recognizable visceral materials: bones, organs, muscle tissue, soft tissue, cells, chemicals, fat and saliva
Visco-elastic properties, properties of visceral organs, measure of stiffnesss

FlanPRF Input Query: Improve the search effectiveness by suggesting expansion terms for the query: define visceral, based on the given context information: Abdominal fat can be visceral or subcutaneous. Visceral adipose tissue (VAT) is fat that is around the organs inside the abdomen. Subcutaneous adipose tisue (SAT) is fat under your skin (love handles). With visceral fat your stomach wall is pushed out from inside. Your stomach muscles can sometimes be quite defined, but your stomach will still be extended. isceral adipose tissue (VAT) is fat that is around the organs inside the abdomen. Subcutaneous adipose tisue (SAT) is fat under your skin (love handles). With visceral fat your stomach wall is pushed out from inside. Your stomach muscles can sometimes be quite defined, but your stomach will still be extended.
FlanPRF Reformulated Query:
Visceral adipose tissue (VAT) is fat that is around the organs inside the abdomen.
Visceral (adjective) Abdominal fat can be visceral or subcutaneous.
Abdominal fat can be visceral or subcutaneous. Visceral adipose tissue (VAT) is fat that is around the organs inside the abdomen.
Visceral adipose tissue (VAT) Visceral adipose tissue (VAT) is fat that is around the organs inside the abdomen.
Abdominal fat can be visceral or subcutaneous. Visceral adipose tissue (VAT) is fat that is around the organs inside the abdomen.

Chapter 4

ColBERT-PRF: Semantic PRF for Dense Retrieval

In Chapter 3, we validated our first posed hypothesis in our proposed thesis statement (cf. Section 1.1), namely that pseudo-relevance feedback information can be used by a sequence-to-sequence neural model to generate more effective query reformulations for sparse retrieval. The promising results of using the pretrained language models for query reformulation motivate us to further delve deeper into how to harness the capacity of pretrained language models for effective query reformulation.

Indeed, as discussed in Section 2.2 and Section 2.3, many pretrained language models have demonstrated further promise in being a suitable basis for *dense retrieval*. Typically, instead of using a classical inverted index, in dense retrieval, the documents and queries are represented using embeddings. Then, the documents can be retrieved using an approximate nearest neighbour algorithm – as exemplified by the FAISS toolkit (Johnson et al., 2019). In particular, two distinct families of approaches were introduced in Section 2.3: single representation dense retrieval and multiple representation dense retrieval. In single representation dense retrieval (introduced in Section 2.3.1), as used by DPR and ANCE, each query or document is represented entirely by a single embedding, typically obtained from the BERT's [CLS] token. Query-document relevance is estimated in terms of the similarity of the corresponding [CLS] embeddings. In contrast, in multiple representation dense retrieval (introduced in Section 2.3.2) – as proposed by Khattab and Zaharia (2020) – each term of the queries and documents is represented by a single embedding. For each query embedding, one per query term, the nearest document token embeddings are identified using an approximate nearest neighbour search, before a final re-scoring to obtain exact relevance estimations. Although it has been found that performing information retrieval based on contextualised representations of the query and document can alleviate both the *lexical mismatch*, for instance, "last name" and "surname" and the *semantic mismatch*, for instance, "I like an apple" and "I like Apple airpods" (Peters et al., 2018), user's queries can still be underrepresented. Therefore, we argue that, as users issue the query prior to access to the relevant documents, the users' queries can still be insufficiently well represented within the dense retrieval paradigm, and as a consequence, this representation can be improved by access to a pseudo-relevant set. Indeed, as introduced in Section 2.2, we know that the BERT model excels at capturing contextualised meanings in textual passages; for instance, the BERT-based dense retrieval models introduced in Section 2.3. However, such dense retrieval models can still fail to interpret the search intent of short user queries, such as 'jaguar'.

On the other hand, many studies have focused on the use of *static* word embeddings, as discussed in Section 2.1,, such as *Word2Vec*, within query expansion methods (Diaz et al., 2016, Kuzi et al., 2016, Roy et al., 2018, 2016). Indeed, most of the existing embedding-based QE methods (Diaz et al., 2016, Kuzi et al., 2016, Roy et al., 2018, 2016, Zamani and Croft, 2016) are based on static embeddings, where a word embedding is always the same within different sentences, and hence they do not address contextualised language models such as BERT. As discussed in Section 2.4.2, CEQE makes use of contextualised BERT embeddings for query expansion. However, the resulting refined query representation again relies on a traditional sparse inverted index for a further round of retrieval. In contrast, in this chapter, we focus on implementing contextualised embedding-based query expansion for dense retrieval. Accordingly, we pose the follow-up question corresponding to the second hypothesis of our thesis statement in Section 1.1: *can we apply the pseudo-relevance feedback mechanism on contextualised embeddings to refine the query representation for multiple representation dense retrieval?*

Indeed, as retrieval uses multiple representations, this allows additional useful embeddings to be appended to the query representation. Furthermore, the exact scoring stage provides the document embeddings in response to the original query, which can be used as pseudo-relevance information. In particular, we propose a pseudo-relevance feedback mechanism called ColBERT-PRF for dense retrieval. More specifically, as embeddings cannot be counted, ColBERT-PRF applies clustering to the embeddings occurring in the pseudo-relevant set, and then identifies the most discriminative embeddings among the cluster centroids. These centroids are then appended to the embeddings of the original query. ColBERT-PRF is focussed on multiple representation dense retrieval settings; However, compared to existing work, our approach is the first work to apply pseudo-relevance feedback to any form of dense retrieval setting; moreover, among the existing approaches applying deep learning for pseudo-relevance feedback, our work in this paper is the first that can improve the recall of the candidate set by re-executing the expanded query representation upon the dense retrieval index, and thereby identify more relevant documents that can be highly ranked for the user.

To summarise, this chapter makes the following contributions:

- We propose a novel contextualised pseudo-relevance feedback mechanism for multiple representation dense retrieval;
- We cluster and rank the feedback document embeddings for selecting candidate expansion embeddings;

- We evaluate our proposed contextualised PRF model in both ranking and reranking settings.
- We demonstrate the effectiveness of the ColBERT-PRF model on document ranking tasks, using the MSMARCO document test collection and the TREC Robust04 test collections;
- We further investigate the effectiveness of ColBERT-PRF by varying the selection of the expansion embeddings.
- We thoroughly investigate the trade-off between the effectiveness and the efficiency of ColBERT-PRF.

The remainder of this chapter is organised as follows. In Section 4.1, we review and provide a more detailed description of a multi-representation dense retrieval, the ColBERT model. Section 4.2 presents our proposed dense PRF method – ColBERT-PRF. Next, we discuss the effectiveness of ColBERT-PRF for the passage ranking task and for the document ranking task in Section 4.3 and Section 4.4, respectively. Next, we discuss the usefulness of different weighting methods for measuring the informativeness of the expansion embeddings of ColBERT-PRF in Section 4.5. In Section 4.6, we study efficient variants of ColBERT-PRF. Finally, we provide concluding remarks of this chapter in Section 4.7.

4.1 Recap: Multi Representation Dense Retrieval

As introduced in Section 2.3.2, in ColBERT, the queries and documents are represented by tokens from a vocabulary V. Each token occurrence has a contextualised real-valued vector with dimension d, called an embedding. More formally, let $f : V^n \rightarrow \mathbb{R}^{n\times d}$ be a function mapping a sequence of terms $\{t_1,\ldots,t_n\}$, representing a query q, composed by $|q|$ tokens into a set of embeddings $\{\phi_{q_1},\ldots,\phi_{q_{|q|}}\}$ and a document composed by $|d|$ tokens into a set of embeddings $\{\phi_{d_1},\ldots,\phi_{d_{|d|}}\}$.

Khattab and Zaharia (2020) recommended that the number of query embeddings be 32, with extra [MASK] tokens being used as query augmentation. Indeed, these mask tokens are a differentiable mechanism that allows documents to gain score contributions from embeddings that do not actually occur in the query, but which the model assumes could be present in the query. In practice, as we later show in Section 4.2.4, the [MASK] embeddings are very similar to embeddings of the existing query tokens, and hence cannot be considered as a form of query expansion. Moreover, they do not make use of pseudo-relevance feedback information obtained from the top-ranked documents of the original query, which has repeatedly been shown to be an effective source to improve query representations.

To obtain the first set of candidate documents, ColBERT makes use of FAISS in its first-stage pass, an approximate nearest neighbour search library, on the pre-computed document embeddings. Conceptually, FAISS allows to retrieve the k' documents containing the nearest neighbour

Algorithm 1: The ColBERT E2E algorithm

```
Input  : A query $Q$
Output : A set $A$ of (docid, score) pairs
ColBERT E2E ($Q$) :
    $\phi_{q_1}, \ldots, \phi_{q_n} \leftarrow \mathsf{Encode}(Q)$
    $D \leftarrow \emptyset$
    for $\phi_{q_i}$ in $\phi_{q_1}, \ldots, \phi_{q_n}$ do
        $D \leftarrow D \cup \mathcal{F}_d(\phi_{q_i}, k')$
    $A \leftarrow \emptyset$
    for $d$ in $D$ do
        $s \leftarrow \sum_{i=1}^{|q|} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}$
        $A \leftarrow A \cup \{(d, s)\}$
    return $A$
```

document embeddings to a query embedding ϕ_{q_i}, i.e., it provides a function $\mathcal{F}_d(\phi_{q_i}, k') \rightarrow (d, \ldots)$ that returns a list of k' documents, sorted in decreasing approximate scores.

However, these approximate scores are insufficient for accurately depicting the similarity scores of the documents, hence the accurate final document scores are computed using Equation (2.21) in a second pass. Typically, for each query embedding, the nearest $k' = 1{,}000$ documents are identified. The set formed by the union of these documents are reranked[1] using Equation (2.21). A separate index data structure (typically in memory) is used to store the uncompressed embeddings for each document.

To the best of our knowledge, ColBERT exemplifies the implementation of an end-to-end IR system that uses multiple representation. Algorithm 1 summarises the ColBERT retrieval algorithm for the end-to-end dense retrieval introduced in Section 2.3.2.

The easy access to the document embeddings used by ColBERT provides an excellent basis for our dense retrieval pseudo-relevance feedback approach. Indeed, while the use of embeddings in ColBERT addresses the vocabulary mismatch problem, we argue that identifying more related embeddings from the top-ranked documents may help to further refine the document ranking. In particular, as we will show, this permits representative embeddings from a set of pseudo-relevance documents to be used to refine the query representation ϕ.

4.2 Dense Pseudo-Relevance Feedback

The aim of a pseudo-relevance feedback approach is typically to generate a refined query representation by analysing the text of the feedback documents. In our proposed ColBERT-PRF approach, we are inspired by conventional PRF approaches, as discussed in Section 2.4.1, such

[1] In this way, any notion of similarity from the ANN stage is discarded - the entire set of retrieved documents is reranked; we return to this detail later in Section 4.6.

as Bo1 and RM3, which assume that good expansion terms will occur frequently in the feedback set (and hence are somehow *representative* of the information need underlying the query), but infrequent in the collection as a whole (therefore are sufficiently *discriminative*). Therefore, we aim to encapsulate these intuitions while operating in the contextualised embedding space $\mathbb{R}^d$, where the exact counting of frequencies is not actually possible. In particular, by operating entirely in the embedding space rather than directly on tokens, we conjecture that we can identify similar embeddings (corresponding to tokens with similar contexts), which can be added to the query representation for improved effectiveness.[2]

In this section, we detail how we identify representative (centroid) embeddings from the feedback documents (Section 4.2.1), how we ensure that those centroid embeddings are sufficiently discriminative (Section 4.2.2), and how we apply these discriminative representative centroid embeddings for (re)ranking (Section 4.2.3). We conclude with an illustrative example (Section 4.2.4) and a discussion of the novelty of ColBERT-PRF (Section 4.2.5).

4.2.1 Representative Embeddings in Feedback Documents

First, we need to identify representative embeddings $\{\upsilon_1, \ldots, \upsilon_K\}$ among all embeddings in the feedback documents set. A typical "sparse" PRF approach – such as RM3 – would count the frequency of terms occurring in the feedback set to identify representative ones. However, in a dense embedded setting, the document embeddings are not countable. Instead, we resort to clustering to identify patterns in the embedding space that are representative of embeddings.

Specifically, let $\Phi(q, f_b)$ be the set of all document embeddings from the f_b top-ranked feedback documents. Then, we apply a clustering approach, e.g., the KMeans clustering algorithm, to $\Phi(q, f_b)$:

$$\{\upsilon_1, .., \upsilon_K\} = \mathsf{Clustering}\big(K, \Phi(q, f_b)\big). \tag{4.1}$$

By applying the clustering algorithm, we obtain K representative centroid embeddings of the feedback documents. The embeddings forming each cluster may or may not correspond to the exact same tokens spread across the feedback documents. In this way, a cluster can represent one or more tokens that appear in similar contexts, rather than a particular exact token. This is a key advantage of ColBERT-PRF. For example, for the query 'do goldfish grow?' (discussed further in Section 4.2.4 below), we find that the embeddings for the words 'water', 'tank', 'tanks' and 'pond' form one of the clusters; other clusters are for unrelated words, such as 'to', 'change', 'color' etc. In the next section, we address the selection of centroids, specifically identifying those that discriminate among the documents in the corpus, and hence should be used for expansion.

To further demonstrate the choice of clustering technique for ColBERT-PRF, we have compared ColBERT-PRF implemented using KMeans clustering and ColBERT-PRF with traditional

[2] In (Wang et al., 2022c), we provide experiments that use Bo1 and RM3 to select tokens and their corresponding embeddings that verify this conjecture.

query expansion methods, namely Bo1 and RM3 techniques in (Wang et al., 2022c). Later, in Section 4.6, we propose and evaluate other approaches for clustering.

4.2.2 Identifying Discriminative Embeddings among Representative Embeddings

Many of the K representative embeddings may represent stopwords and therefore are not sufficiently informative when retrieving documents. Typically, identifying informative and discriminative expansion terms from feedback documents would involve examining the collection frequency or the document frequency of the constituent terms (Cao et al., 2008, Roy et al., 2019). However, there may not be a one-to-one relationship between query/centroid embeddings and actual tokens, hence we seek to map each centroid υ_i to a possible token t. Some clusters will be focused in their meaning, containing only a single token (but with multiple different embeddings) or containing embeddings of multiple related tokens (e.g. 'water', 'tank', 'tanks' and 'pond'). Other clusters will be less focused, containing unrelated words, such as non-functional words and stopwords ('to', 'change', 'color', 'from', 'or', 'were').

To map each cluster to a single token to represent each cluster, we resort to FAISS, through the function $\mathcal{F}_t(\upsilon_i, r) \rightarrow (t, \ldots)$ that, given the centroid embedding υ_i and r, returns the list of the r *token ids* corresponding to the r closest document embeddings to the centroid.[3] From a probabilistic viewpoint, the likelihood $P(t|\upsilon_i)$ of a token t given an embedding υ_i can be obtained as:

$$P(t|\upsilon_i) = \frac{1}{r} \sum_{\tau \in \mathcal{F}_t(\upsilon_i, r)} \mathbb{1}[\tau = t], \tag{4.2}$$

where $\mathbb{1}[]$ is the indicator function.

For simplicity, we choose the most likely token id, i.e., $t_i = \arg\max_t P(t|\upsilon_i)$. Mapping back to a token id allows us to make use of Inverse Document Frequency (IDF), which can be pre-recorded for each token ID. The importance σ_i of a centroid embedding υ_i is obtained using a traditional IDF formula[4]: $\sigma_i = \log\left(\frac{N+1}{N_i+1}\right)$, where N_i is the number of passages containing the token t_i and N is the total number of passages in the collection. While this approximation of embedding informativeness is obtained by mapping back to tokens, as we shall show, it is very effective.

Choosing the single most likely token from FAISS is useful for two reasons: while a cluster may encapsulate multiple tokens, using FAISS tells us what the cluster's centroid embedding will actually match to in the corpus; Moreover, a qualitative inspection looking at either the mostly likely tokens identified from FAISS, or the tokens involved in each cluster, found that in general they were representative of the cluster in terms of informativeness. For instance, a cluster containing embeddings for 'to', 'change', 'color', 'from', 'or', 'were', the centroid was most likely

[3] This additional mapping can be recorded at indexing time, using the same FAISS index as for dense retrieval, increasing the index size by 3%. [4] We have observed no marked empirical benefits in using other IDF formulations.

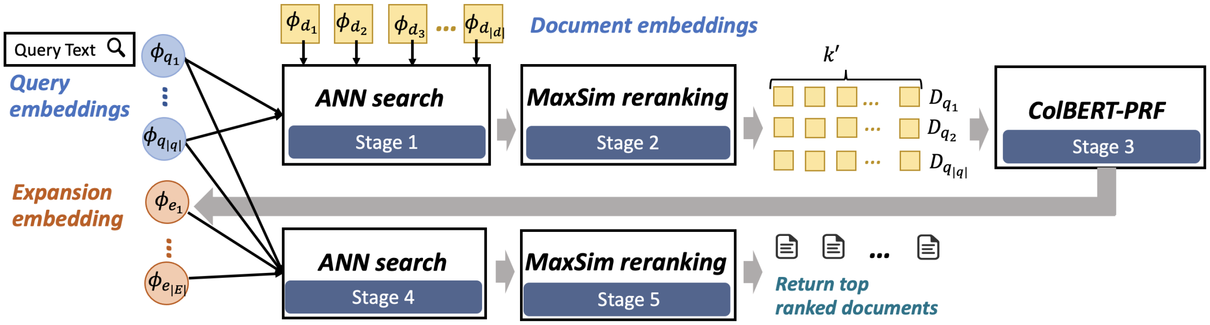


Figure 4.1: Workflow of ColBERT-PRF ranker.

to match with ‘to’ in the corpus, meaning that the centroid was not useful for expansion. Going further, instead of IDF, we discuss different derivations of a tailored informativeness measure in Section 4.5, including Inverse Collection Term Frequency and Mean Cosine Similarity methods.

Finally, we select the f_e most informative centroids as expansion embeddings based on the σ_i importance scores as follows:

$$F_e = \mathsf{TopScoring}\Big(\big\{(\upsilon_1,\sigma_1),\ldots,(\upsilon_K,\sigma_K)\big\}, f_e\Big), \tag{4.3}$$

where $\mathsf{TopScoring}(A,c)$ returns the c elements of A with the highest importance score.

4.2.3 Ranking and Reranking with ColBERT-PRF

Given the original $|q|$ query embeddings and the f_e expansion embeddings, we incorporate the score contributions of the expansion embeddings in Eq. (2.21) as follows:

$$s(q,d) = \sum_{i=1}^{|q|} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j} + \beta \sum_{(\upsilon_i,\sigma_i)\in F_e} \max_{j=1,\ldots,|d|} \sigma_i \upsilon_i^T \phi_{d_j}, \tag{4.4}$$

where $\beta > 0$ is a parameter weighting the contribution of the expansion embeddings, and the score produced by each expansion embedding is further weighted by the IDF weight of its most likely token, σ_i. Note that Equation (4.4) can be applied to rerank the documents obtained from the initial query, or as part of a full re-execution of the full dense retrieval operation including the additional f_e expansion embeddings.

In both ranking and reranking, ColBERT-PRF has 4 parameters: f_b, the number of feedback documents; K, the number of clusters; $f_e \leq K$, the number of expansion embeddings; and β, the importance of the expansion embeddings during scoring. Figure 4.1 presents the five stages of ColBERT-PRF in its ranking configuration.

Furthermore, we provide the pseudo-code of our proposed ColBERT PRF ReRanker in Algorithm 2. The ColBERT-PRF Ranker can be easily obtained by inserting lines 3-4 of Algorithm 1 at line 10 of Algorithm 2 to perform retrieval using both the original query embeddings and the expansion embeddings, and similarly adapting the MaxSim scoring in Eq. (2.21) to encapsulate the original query embeddings as well as the expansion embeddings.

Finally, we demonstrate the stages of ColBERT-PRF, for both ColBERT-PRF Ranker and ReRanker scenarios, when defined as PyTerrier pipelines (see the notations introduced in Section 2.1.2). In Listing 1, we portray the experimental pipelines for ColBERT E2E and ColBERT-PRF. The original source code can be found in the PyTerrier_ColBERT repository.[5]

Listing 1: ColBERT-PRF Pipeline.

```
# Loading the ColBERT index
from pyterrier_colbert.ranking import ColBERTFactory
pytcolbert = ColBERTFactory("/path/to/checkpoint.dnn",
                            "/path/to/index", "index_name")
# Build the experimental pipeline
def prf(pytcolbert, rerank, fb_docs=3, fb_embs=10, beta=1.0, k=24) -> Transformer:
    # Pipeline for ColBERT E2E: dense_e2e
    dense_e2e = (pytcolbert.set_retrieve()
              >> pytcolbert.index_scorer(query_encoded=True, add_ranks=True,
                                        batch_size=10000))
    if rerank:
        # Build pipeline for ColBERT-PRF ReRanker
        prf_pipe = (
                dense_e2e
                >> ColbertPRF(pytcolbert, k=k, fb_docs=fb_docs,
                              fb_embs=fb_embs, beta=beta, return_docs=True)
                >> (pytcolbert.index_scorer(query_encoded=True,
                                            add_ranks=True,
                                            batch_size=5000) %1000)
            )
    else:
        # Build pipeline for ColBERT-PRF Ranker
        prf_pipe = (
                dense_e2e
                >> ColbertPRF(pytcolbert, k=k, fb_docs=fb_docs,
                              fb_embs=fb_embs, beta=beta, return_docs=False)
                >> pytcolbert.set_retrieve(query_encoded=True)
                >> (pytcolbert.index_scorer(query_encoded=True,
                                            add_ranks=True,
                                            batch_size=5000) % 1000)
             )
    return prf_pipe
```

[5] http://github.com/terrierteam/pyterrier_colbert

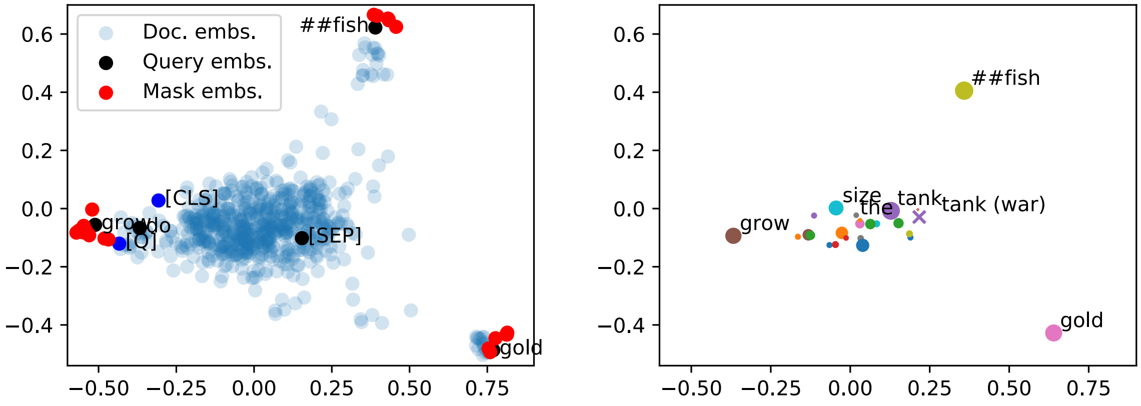
Algorithm 2: The ColBERT PRF (reranking) algorithm

```
Input  : A query $Q$,
         number of feedback documents $f_b$,
         number of representative embeddings $K$,
         number of expansion embeddings $f_e$
Output : A set $B$ of (docid, score) pairs
ColBERT PRF ($Q$) :
    $A \leftarrow$ ColBERT E2E($Q$)
    $\Phi(Q, f_b) \leftarrow$ set of all document embeddings from
          the $f_b$ top-scored documents in $A$
    $V \leftarrow \emptyset$
    $\upsilon_1, \ldots, \upsilon_K$ = KMeans$\big(K, \Phi(Q, f_b)\big)$
    for $\upsilon_i$ in $\upsilon_1, \ldots, \upsilon_K$ do
        $t_i \leftarrow \arg\max_t \frac{1}{r} \sum_{\tau \in \mathcal{F}_t(\upsilon_i, r)} \mathbb{1}[\tau = t]$
        $\sigma_i \leftarrow \log\left(\frac{N+1}{N_i+1}\right)$
        $V \leftarrow V \cup \big\{(\upsilon_i, \sigma_i)\big\}$
    $F_e \leftarrow$TopScoring($V, f_e$)
    $B \leftarrow \emptyset$
    for $(d, s)$ in $A$ do
        $s \leftarrow s + \beta \sum_{(\upsilon_i, \sigma_i) \in F_e} \max_{j=1,\ldots,|d|} \sigma_i \upsilon_i^T \phi_{d_j}$
        $B \leftarrow B \cup \big\{(d, s)\big\}$
    return $B$
```

4.2.4 Illustrative Example

We now illustrate the effect of ColBERT-PRF upon one query from the TREC 2019 Deep Learning track, ‘do goldfish grow’. We use PCA to quantize the 128-dimension embeddings into 2 dimensions purely to allow visualisation. Firstly, Figure 4.2(a)shows the embeddings of the original query (black ellipses); the red [MASK] tokens are also visible, clustered around the original query terms (##fish, gold, grow). Meanwhile, document embeddings extracted from 10 feedback documents are shown as light blue ellipses in Figure 4.2(a). There appear to be visible clusters of document embeddings near the query embeddings, but also other document embeddings exhibit some clustering. The mass of embeddings near the origin is not distinguishable in PCA. Figure 4.2(b) demonstrates the application of KMeans clustering upon the document embeddings; we map back to the original tokens by virtue of Equation (4.2). In Figure 4.2(b), the point size is indicative of the IDF of the corresponding token. We can see that the cluster centroids with high IDF correspond to the original query tokens (‘gold’, ‘##fish’, ‘grow’), as well as the related terms (‘tank’, ‘size’). In contrast, a centroid with low IDF is ‘the’. This illustrates the utility of our proposed ColBERT-PRF approach in using KMeans to identify representative clusters of embeddings, as well as using IDF to differentiate useful clusters.

Furthermore, Figure 4.2(b) also includes, marked by an × and denoted ‘tank (war)’, the



(a) Query & doc. embeddings.

(b) Cluster centroids, $K = 24$.

Figure 4.2: Example showing how ColBERT-PRF operates for the query ‘do goldfish grow’ in a 2D PCA space. In Figure 4.2(b), the point size is representative of IDF; five high IDF and one low IDF centroids are shown. For contrast, × ‘tank (war)’ denotes the embedding of ‘tank’ occurring in a non-fish context.

embedding for the word ‘tank’ when placed in the passage *“While the soldiers advanced, the tank bombarded the troops with artillery”*. It can be seen that, even in the highly compressed PCA space, the ‘tank’ centroid embedding is distinct from the embedding of ‘tank (war)’. This shows the utility of ColBERT-PRF when operating in the embedding space, as the PRF process for the query ‘do goldfish grow’ will not retrieve documents containing ‘tank (war)’, but will focus on a fish-related context, thereby dealing with the polysemous nature of a word such as ‘tank’. To the best of our knowledge, this is a unique feature of ColBERT-PRF among PRF approaches.

4.2.5 Discussion

To the best of our knowledge ColBERT-PRF is the first investigation of pseudo-relevance feedback for multiple representation dense retrieval. Existing works on neural pseudo-relevance feedback that have been introduced in Section 2.4.2, such as Neural PRF and BERT-QE only function as rerankers. Other approaches such as DeepCT and doc2query use neural models (cf. Section 2.2.3) to augment documents before indexing using a traditional inverted index. CEQE (cf. Section 2.4.2) generates words to expand the initial query, which is then executed on the inverted index. However, returning the BERT embeddings back to textual word forms can result in polysemous words negatively affecting retrieval. In contrast, ColBERT-PRF operates entirely on an existing dense index representation (without augmenting documents), and can function for both ranking as well as reranking. By retrieving using feedback embeddings directly, ColBERT-PRF addresses polysemous words (such as ‘tank’, illustrated above). It is also of note that it also requires no additional neural network training beyond that of ColBERT. Indeed, while ANCE-PRF (cf. Section 2.4.3) requires further training of the refined query encoder, ColBERT-PRF does not require any further retraining. Furthermore, compared to the single embedding of ANCE-PRF, ColBERT-PRF is also more explainable in nature, as the expansion embeddings can be mapped to tokens (as shown in Figure 4.2), and their contribution to document scoring can be examined, as we will show in Section 4.3.3.4.

In the following, we first show the retrieval effectiveness of ColBERT-PRF for passage ranking and document ranking tasks in Section 4.3 and Section 4.4, respectively. In particular, in Section 4.3, we examine the characteristics of ColBERT-PRF, including how ColBERT-PRF addresses polysemous words, how ColBERT-PRF demonstrates compared with the traditional query expansion techniques and how to quantify the extent of the semantic matching ability of ColBERT-PRF. Next, we discuss three variants of ColBERT-PRF with different discriminative power measure methods in Section 4.5, and we address the effectiveness and efficiency trade-off of ColBERT-PRF in Section 4.6.

4.3 Passage Ranking Effectiveness of ColBERT-PRF

In this section, we analyse the performance of ColBERT-PRF for passage ranking. In particular, we evaluated the performance of ColBERT-PRF on TREC 2019 and TREC 2020 query sets. Section 4.3.1 describes the research question addressed by our passage ranking experiments. The experimental setup and the obtained results are detailed in Section 4.3.2 and Section 4.3.3, respectively.

4.3.1 Research Questions

Our passage ranking experiments address the four following research questions:

- RQ4.1 Can a multiple representation dense retrieval approach be enhanced by pseudo-relevance feedback, i.e., can ColBERT-PRF outperform ColBERT dense retrieval?
- RQ4.2 How does ColBERT-PRF compare to other existing baselines and state-of-the-art approaches, namely:
  (a) lexical (sparse) baselines, including using PRF,
  (b) neural augmentation approaches, namely DeepCT and docT5query,
  (c) BERT-QE Reranking models,
  (d) embedding based query expansion models, namely the three variants of CEQE models: CEQE-Max, CEQE-Centroid and CEQE-Mul?
- RQ4.3 What is the impact of the parameters of ColBERT-PRF, namely the number of clusters and expansion embeddings, the number of feedback passages and the β parameter controlling the influence of the expansion embeddings?
- RQ4.4 To what extent does ColBERT-PRF perform semantic matching?

4.3.2 Experimental Setup

Now we introduce the experimental setup for the evaluation of ColBERT-PRF on the passage ranking task. More specifically, the dataset and metrics are introduced in Section 4.3.2.1, then the implementation details and baselines are detailed in Sections 4.3.2.2 and 4.3.2.3, respectively.

4.3.2.1 Dataset & Measures

Experiments are conducted on the MSMARCO passage corpus, using 43 TREC 2019 DL track topics and 54 TREC 2020 DL track topics (cf. Section 2.5.1).

We perform the evaluation on the TREC 2019 and TREC 2020 query sets using the metrics introduced in Section 2.5, namely the mean reciprocal rank (MRR) and normalised discounted cumulative gain (nDCG) calculated at rank cutoff 10, as well as Recall and Mean Average Prevision (MAP) at rank 1000. For the MRR, MAP and Recall metrics, we treat passages with label grade 1 as non-relevant, following (Craswell et al., 2021a,b). In addition, following the efficiency metrics introduced in Section 2.5, we report the Mean Response Time (MRT) for each retrieval system. For significance testing, we use the paired t-test ($p < 0.05$) and apply the Holm-Bonferroni multiple testing correction.

4.3.2.2 Implementation and Settings

We conduct experiments using PyTerrier (cf. Section 2.1.2) and, in particular using our PyTerrier_ColBERT plugin[6], which includes ColBERT-PRF as well as our adaptations of the ColBERT source code. ColBERT and ColBERT-PRF are expressed as PyTerrier transformer operations. In addition, the ColBERT-PRF and the baselines results are available in our virtual appendix[7].

In terms of the ColBERT configuration, we train ColBERT upon the MSMARCO passage ranking triples file for 44,000 batches, applying the parameters specified by Khattab & Zaharia in (Khattab and Zaharia, 2020): Maximum document length is set to 180 tokens and queries are encoded into 32 query embeddings (including [MASK] tokens); We encode all passages to a FAISS index that has been trained using 5% of all embeddings; At retrieval time, FAISS retrieves $k' = 1000$ passage embeddings for every query embedding. ColBERT-PRF is implemented using the KMeans implementation (Arthur and Vassilvitskii, 2007) of sci-kit learn (sklearn). For query expansion settings, we follow the default settings of Terrier (Ounis et al., 2005), which is 10 expansion terms obtained from 3 feedback passages; we follow the same default setting for ColBERT-PRF, additionally using representative values, namely $K = 24$ clusters[8], and $\beta = \{0.5, 1\}$ for the weight of the expansion embeddings. We later show the impact of these parameters when we address RQ4.3.

[6] `github.com/terrierteam/pyterrier_colbert` [7] https://github.com/Xiao0728/ColBERT-PRF-VirtualAppend

[8] Indeed, $K = 24$ gave reasonable looking clusters in our initial investigations, and, as we shall see in Section 6.3, is an effective setting for the TREC 2019 query set.

4.3.2.3 Baselines

To test the effectiveness of our proposed dense PRF approach, we compare with five families of baseline models, for which we vary the use of a BERT-based reranker (namely BERT or ColBERT). For the BERT reranker, we use OpenNIR (MacAvaney, 2020) and `capreolus/bert-base-msmarco` fine-tuned model from (Li et al., 2020). For the ColBERT reranker, unless otherwise noted, we use the existing pre-indexed ColBERT representation of passages for efficient reranking. The five families are:

- **Lexical Retrieval Approaches:** These are traditional retrieval models using a sparse inverted index, which are introduced in Section 2.1.1, with and without BERT and ColBERT rerankers, namely: (i) BM25 (ii) BM25+BERT (iii) BM25+ColBERT, (iv) BM25+RM3, (v) BM25+RM3+BERT and (vi) BM25+RM3+ColBERT.

- **Neural Augmentation Approaches:** These use neural components to augment the (sparse) inverted index that have been discussed in Section 2.2.3: (i) BM25+DeepCT and (ii) BM25+docT5query, both without and with BERT and ColBERT rerankers. For BM25+docT5query+ColBERT, the ColBERT reranker is applied on expanded passage texts encoded at querying time, rather than the indexed ColBERT representation. The response time for BM25+docT5query+ColBERT reflects this difference.

- **Dense Retrieval Models:** This family consists of the dense retrieval approaches: (i) ANCE (cf. Section 2.3.1): The ANCE model is a single representation dense retrieval model. We use the trained models provided by the authors trained on MSMARCO training data. (ii) ANCE-PRF (cf. Section 2.4.3): The ANCE-PRF is a PRF variant of ANCE model – we use the results released by the authors.(iii) ColBERT E2E (cf. Section 2.3.2): ColBERT end-to-end (E2E) is the dense retrieval version of ColBERT, as defined in Section 4.1.

- **BERT-QE Models:** We apply BERT-QE (cf. Section 2.4.2 on top of a strong sparse baseline and our dense retrieval baseline, ColBERT E2E, i.e., (i) BM25+RM3+ColBERT+BERT-QE and (ii) ColBERT E2E+BERT-QE; Where possible, we use the ColBERT index for scoring passages; for identifying the top scoring chunks within passages, we use ColBERT in a slower “text” mode, i.e., without using the index. For the BERT-QE parameters, we follow the settings in (Zheng et al., 2020), in particular using the recommended settings of $\alpha = 0.4$ and $\beta = 0.9$, which are also the most effective on MSMARCO. Indeed, to the best our knowledge, this is the first application of BERT-QE upon dense retrieval, the first application of BERT-QE on MSMARCO and the first application using ColBERT. We did attempt to apply BERT-QE using the BERT re-ranker, but we found it to be ineffective on MSMARCO, and exhibiting a response time exceeding 30 seconds per query, hence we omit it from our experiments.

- **CEQE Models:** As discussed in Section 2.4.2, this family consists of three CEQE variants, i.e., CEQE-Max, CEQE-Centroid, and CEQE-Mul. We apply each CEQE query expansion variant on top of the documents retrieved by BM25. Compared with the original CEQE, we apply the pipeline BM25 + RM3 + BM25 rather than the Dirichlet LM + RM3 + BM25 pipeline for generating the expansion terms.

4.3.3 Passage Ranking Results

Now we analyse the evaluation performance of our proposed ColBERT-PRF model for the passage ranking task and answer each of the posed research questions from Section 4.3.3.1 to Section 4.3.3.4.

4.3.3.1 RQ4.1 – Overall Effectiveness of ColBERT-PRF

In this section, we examine the effectiveness of a pseudo-relevance feedback technique for the ColBERT dense retrieval model on passage ranking task. As discussed in Section 2.5.3, this set of experiments can be described as the in-domain evaluation as the evaluation and training queries are from the same domain. On analysing Table 4.1, we first note that the ColBERT dense retrieval approach outperforms the single representation based dense retrieval models, i.e., ANCE and its PRF variant ANCE-PRF for all metrics on both test query sets, probably because the single representation used in ANCE provides limited information for matching queries and documents (Luan et al., 2021). In particular, compared with ANCE-PRF, ColBERT-PRF shows markedly improvement on all metrics for both query sets and shows significant improvement in terms of MAP on TREC 2019 and nDCG@10 on TREC 2020. This indicates that the PRF mechanism that explicitly expands query with expansion embeddings to refine the query representation is superior to implicitly learning from PRF information to form a better query representation.

Based on this, we then compare the performances of our proposed ColBERT-PRF models, instantiated as ColBERT-PRF Ranker & ColBERT-PRF ReRanker, with the more effective ColBERT E2E model. We find that both the Ranker and ReRanker models outperform ColBERT E2E on all the metrics for both used query sets. Typically, on the TREC 2019 test queries, both the Ranker and ReRanker models exhibit significant improvements in terms of MAP over the ColBERT E2E model. In particular, we observe a 26% increase in MAP on TREC 2019[9] and 10% for TREC 2020 over ColBERT E2E for the ColBERT-PRF Ranker. In addition, both ColBERT-PRF Ranker and ReRanker exhibit significant improvements over ColBERT E2E in terms of nDCG@10 on TREC 2019 queries.

The high effectiveness of ColBERT-PRF Ranker (which is indeed higher than ColBERT-PRF ReRanker) can be explained in that the expanded query obtained using the PRF process introduces more relevant passages, thus it increases recall after re-executing the query on

[9] Indeed, this is 8% higher than the highest MAP among all TREC 2019 participants (Craswell et al., 2021a).

the dense index. As can be seen from Table 4.1, ColBERT-PRF Ranker exhibits significant improvements over both ANCE and ColBERT E2E models on Recall. On the other hand, the effectiveness of ColBERT-PRF ReRanker also suggests that the expanded query provides a better query representation, which can better rank documents in the existing candidate set. Overall, in response to RQ4.1, we conclude that our proposed ColBERT-PRF model is effective compared to the ColBERT E2E dense retrieval model.

Table 4.1: Comparison with baselines. Superscripts a...p denote significant improvements over the indicated baseline model(s). The highest value in each column is boldfaced. The higher MRT of BM25+docT5query+ColBERT is expected, as we do not have a ColBERT index for the docT5query representation.

	TREC 2019 (43 queries)					TREC 2020 (54 queries)				
	MAP	nDCG@10	MRR@10	Recall	MRT	MAP	nDCG@10	MRR@10	Recall	MRT
Lexical Retrieval Approaches										
BM25 (a)	0.2864	0.4795	0.6416	0.7553	133	0.2930	0.4936	0.5912	0.8103	129
BM25+BERT (b)	0.4441	0.6855	0.8295	0.7553	3589	0.4699	0.6716	0.8069	0.8103	3554
BM25+ColBERT (c)	0.4582	0.6950	0.8580	0.7553	202	0.4752	0.6931	0.8546	0.8103	203
BM25+RM3 (d)	0.3108	0.5156	0.6093	0.7756	201	0.3203	0.5043	0.5912	0.8423	248
BM25+RM3+BERT (e)	0.4531	0.6862	0.8275	0.7756	4035	0.4739	0.6704	0.8079	0.8423	4003
BM25+RM3+ColBERT (f)	0.4709	0.7055	0.8651	0.7756	320	0.4800	0.6877	**0.8560**	0.8423	228
Neural Augmentation Approaches										
BM25+DeepCT (g)	0.3169	0.5599	0.7155	0.7321	54	0.3570	0.5603	0.7090	0.8008	64
BM25+DeepCT+BERT (h)	0.4308	0.7011	0.8483	0.7321	3737	0.4671	0.6852	0.8068	0.8008	3719
BM25+DeepCT+ColBERT (i)	0.4416	0.7004	0.8541	0.7321	129	0.4757	0.7071	0.8549	0.8008	141
BM25+docT5query (j)	0.4044	0.6308	0.7614	0.8263	282	0.4082	0.6228	0.7434	0.8456	295
BM25+docT5query+BERT (k)	0.4802	0.7123	0.8483	0.8263	8025	0.4714	0.6810	0.8160	0.8456	3888
BM25+docT5query+ColBERT (l)	0.5009	0.7136	0.8367	0.8263	2362	0.4733	0.6934	0.8021	0.8456	2381
Dense Retrieval Models										
ANCE (m)	0.3715	0.6537	0.8590	0.7571	199	0.4070	0.6447	0.7898	0.7737	179
ANCE-PRF (n)	0.4253	0.6807	0.8492	0.7912	-	0.4452	0.6948	0.8371	0.8148	-
ColBERT E2E (o)	0.4318	0.6934	0.8529	0.7892	581	0.4654	0.6871	0.8525	0.8245	600
BERT-QE Reranking Models										
BM25 + RM3 + ColBERT + BERT-QE (p)	0.4832	0.7179	0.8754	0.7756	1130	0.4842	0.6909	0.8315	0.8423	1595
ColBERT E2E + BERT-QE (q)	0.4423	0.7013	0.8683	0.7892	1261	0.4749	0.6911	0.8315	0.8245	1328
Embedding-based Query Expansion Models										
BM25 + CEQE-Max (r)	0.3453	0.5382	0.6605	0.8277	15656	0.3380	0.5094	0.6132	0.8561	16103
BM25 + CEQE-Centroid (s)	0.3425	0.5345	0.6595	0.8234	14230	0.3302	0.5099	0.6270	0.8540	15432
BM25 + CEQE-Mul (t)	0.3203	0.4987	0.5941	0.8097	15612	0.2999	0.4749	0.5825	0.8447	14887
ColBERT-PRF Models										
ColBERT-PRF Ranker (β=1)	**0.5431**^{abcdghijmnoqrst}	0.7352^{adgrst}	0.8858^{adt}	0.8706^{abhmo}	4103	0.4962^{adgjmrst}	0.6993^{adgrst}	0.8396^{ad}	**0.8892**^{abghlmno}	4150
ColBERT-PRF ReRanker (β=1)	0.5040^{adgmnoqrst}	0.7369^{adgrst}	0.8858^{adt}	0.7961	3543	0.4919^{adgjrst}	0.7006^{adgrst}	0.8396^{ad}	0.8431^{m}	3600
ColBERT-PRF Ranker (β=0.5)	0.5427^{abcdghijmnoqrst}	0.7395^{adgjmrst}	**0.8897**^{adt}	**0.8711**^{abhmo}	4111	**0.5116**^{adgjmnorst}	0.7153^{adgjrst}	0.8439^{ad}	0.8837^{aghlmno}	4155
ColBERT-PRF ReRanker (β=0.5)	0.5026^{adgmnoqrst}	**0.7409**^{adgjmrst}	**0.8897**^{adt}	0.7977	3470	0.5063^{adgjmrst}	**0.7161**^{adgjrst}	0.8439^{ad}	0.8443^{m}	3477

4.3.3.2 RQ4.2 - Comparison to Baselines

Next, to address RQ4.2(a)-(c), we analyse the performances of the ColBERT-PRF Ranker and ColBERT-PRF ReRanker approaches in comparison to different groups of baselines, namely sparse (lexical) retrieval approaches, neural augmented baselines, and BERT-QE.

For RQ4.2(a), we compare the ColBERT-PRF Ranker and ReRanker models with the lexical retrieval approaches. For both query sets, both Ranker and ReRanker provide significant improvements on all evaluation measures compared to the BM25 and BM25+RM3 models. This is mainly due to the more effective contexualised representation employed in the ColBERT-PRF models than the traditional sparse representation used in the lexical retrieval approaches. Furthermore, both ColBERT-PRF Ranker and ReRanker outperform the sparse retrieval approaches when reranked by either the BERT or the ColBERT models – e.g., BM25+(Col)BERT and BM25+RM3+(Col)BERT – on all metrics. In particular, ColBERT-PRF Ranker exhibits marked improvements over the BM25 with BERT or ColBERT reranking approach for MAP on the TREC 2019 queries. This indicates that our query expansion in the contextualised embedding space produces query representations that result in improved retrieval effectiveness. Hence, in answer to RQ4.2(a), we find that our proposed ColBERT-PRF models show significant improvements in retrieval effectiveness over sparse baselines.

To further gauge the extent of improvements brought by the PRF additional information in the sparse retrieval and the dense retrieval paradigms, we compare the amount of performance improvements in terms of MAP for ColBERT-PRF vs. ColBERT, ANCE-PRF vs. ANCE, and BM25+RM3 vs. BM25 in Figure 4.3. We observe that more queries improved, and by a larger margin, by ColBERT-PRF compared to both RM3 and ANCE-PRF. Furthermore, from Figure 4.3, we find that among the failed queries for ColBERT-PRF, most of these queries also failed for the ANCE-PRF and RM3 approaches. These queries are hard queries that may struggle to be improved by a PRF technique. On the other hand, in Table 4.2, we present the number of queries whose performances are improved, unchanged and degraded when comparing a retrieval system with and without a PRF mechanism applied. We find that ColBERT-PRF has the highest number of improved queries and the lowest number of degraded queries. In the bottom half of Table 4.2, we compute Spearman's ρ correlation coefficient between the performance improvements of different PRF methods – a high positive correlation coefficient would be indicative that the two methods demonstrate a similar effect on different types of queries. From Table 4.2, we see that the correlation coefficient between ColBERT-PRF vs. ColBERT and ANCE-PRF vs. ANCE is highest among all the compared pairs (0.41). Overall, this tells us that while there is no strong correlations between the queries improved by applying PRF to each baseline, ColBERT-PRF and ANCE-PRF are the most correlated pair. Indeed, only moderate correlations are observed, showing that the approaches improve different queries. Moreover, from Figure 4.3 we see that ColBERT-PRF improves more queries and with further margin than ANCE-PRF.

For RQ4.2(b), on analysing the neural augmentation approaches, we observe that both the

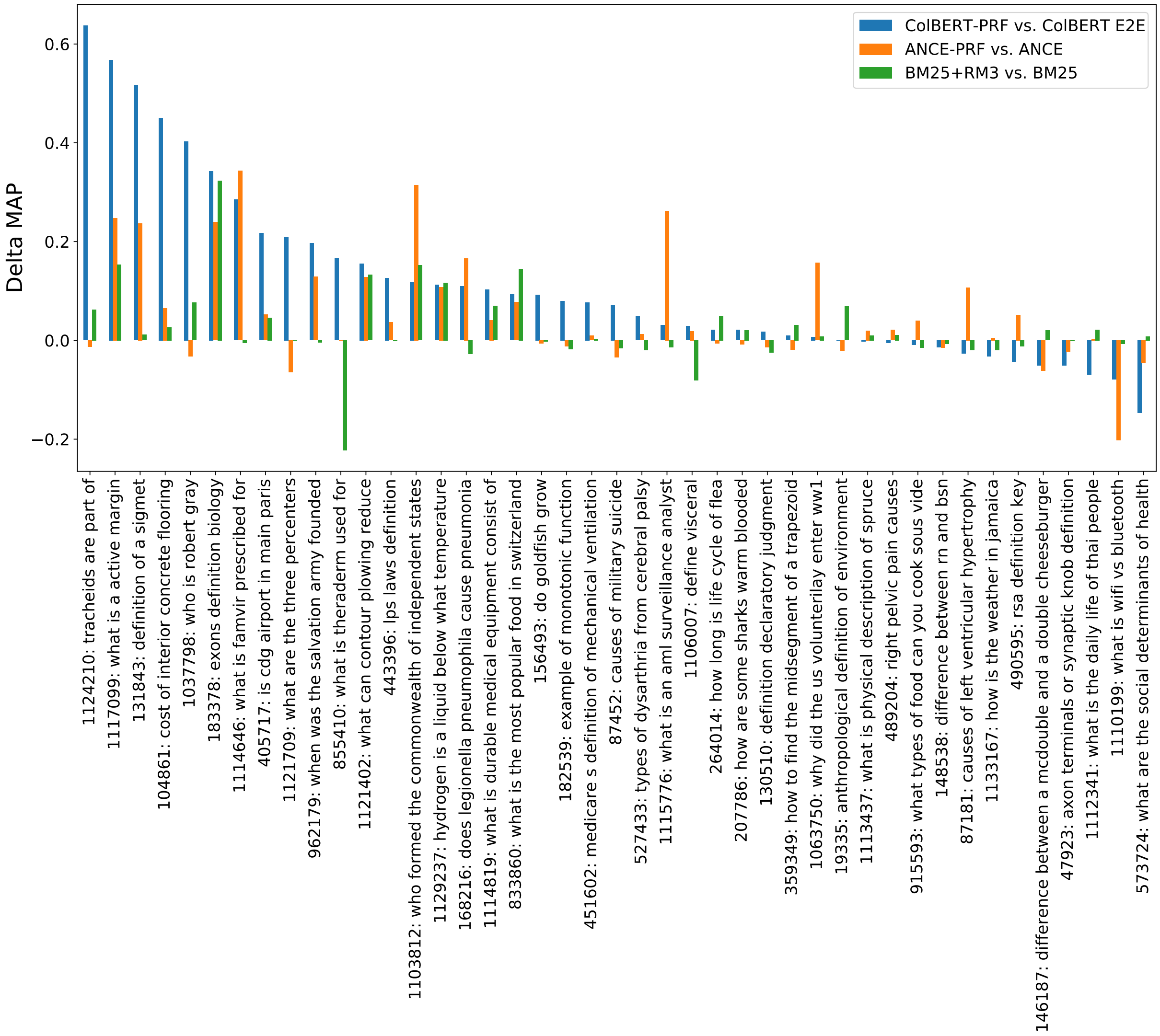


Figure 4.3: Per-query analysis on the TREC 2019 query set.

Table 4.2: Comparison of different PRF mechanisms: (i) numbers of queries improved, unchanged or degraded compared to their respective baselines; (ii) performance improvement correlation (Spearman's ρ correlation coefficient) between pairs of PRF mechanisms.

	BM25+RM3 vs. BM25	ANCE-PRF vs. ANCE	ColBERT-PRF vs. ColBERT E2E
	Improved/Unchanged/Degraded 23/1/19	Improved/Unchanged/Degraded 26/1/16	Improved/Unchanged/Degraded 30/0/13
BM25+RM3 vs. BM25	1.00	0.37	0.34
ANCE-PRF vs. ANCE	0.37	1.00	**0.41**
ColBERT-PRF vs. ColBERT E2E	0.34	**0.41**	1.00

DeepCT and docT5query neural components could lead to effectiveness improvements over the corresponding lexical retrieval models without neural augmentation. However, despite their improved effectiveness, our proposed ColBERT-PRF models exhibit marked improvements over the neural augmentation approaches. Specifically, on the TREC 2019 query set, ColBERT-PRF Ranker significantly outperforms 4 out of 6 neural augmentation baselines and the

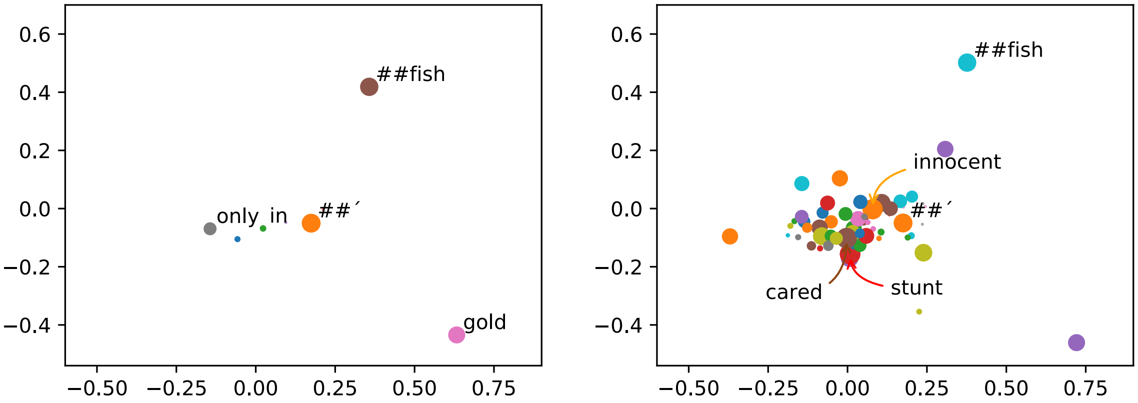
BM25+DeepCT baseline on MAP. Meanwhile, both ColBERT-PRF Ranker and ReRanker exhibit significant improvements over BM25+DeepCT and BM25+docT5query on MAP for TREC 2020 queries, and exhibit improvements upto 9.5% improvements over neural augmentation approaches with neural re-ranking (e.g., MAP 0.4671→0.5116). On analysing these comparisons, the effectiveness of the ColBERT-PRF models indicates that the query representation enrichment in a contextualised embedding space leads to a higher effectiveness performance than the sparse representation passage enrichment. Thus, in response to RQ4.2(b), the ColBERT-PRF models exhibit markedly higher performances than the neural augmentation approaches.

We further compare the ColBERT-PRF models with the recently proposed BERT-QE Reranking model. In particular, we provide results when using BERT-QE to rerank both BM25+RM3 as well as ColBERT E2E. Before comparing the ColBERT-PRF models with the BERT-QE rerankers, we first note that BERT-QE doesn't provide benefit to MAP on either query set, but can lead to a marginal improvement for nDCG@10 and MRR@10. However, the BERT-QE reranker models still underperform compared to our ColBERT-PRF models. Indeed, ColBERT E2E+BERT-QE exhibits a performance significantly lower than both ColBERT-PRF Ranker and ReRanker on the TREC 2019 query set. Hence, in response to RQ4.2(c), we find that the ColBERT-PRF models significantly outperform the BERT-QE reranking models.

Finally, we consider the mean response times reported in Table 4.1, noting that ColBERT PRF exhibits higher response times than other ColBERT-based baselines, and similar to BERT-based re-rankers. There are several reasons for ColBERT PRF's speed: Firstly, the KMeans clustering of the feedback embeddings is conducted online, and the scikit-learn implementation we used is fairly slow – we tried other markedly faster KMeans implementations, but they were limited in terms of effectiveness (particularly for MAP), perhaps due to the lack of the KMeans++ initialisation procedure (Arthur and Vassilvitskii, 2007), which scikit-learn adopts; Secondly ColBERT PRF adds more expansion embeddings to the query - for the ranking setup, each feedback embedding can potentially cause a further $k' = 1000$ passages to be scored - further tuning of ColBERT's k' parameter may allow efficiency improvements for ColBERT-PRF without much loss of effectiveness, at least for the first retrieval stage.Based on this, we further investigate how to attain more of a balance between the effectiveness and the efficiency in leveraging techniques such as approximate scoring technique (Macdonald and Tonellotto, 2021) and other clustering algorithms.

4.3.3.3 RQ4.3 - Impact of ColBERT-PRF Parameters.

To address RQ4.3, we investigate the impact of the parameters of ColBERT-PRF. In particular, when varying the values of a specific hyper-parameter type, we fix all the other hyper-parameters to their default setting, i.e. $f_b = 3$, $f_e = 10$, $\beta = 1$ and $k = 24$. Firstly, concerning the number of clusters, K, and the number of expansion embeddings f_e selected from those clusters ($f_e \leq K$), Figures 4.5(a) and (b) report, for ColBERT-PRF Ranker and ColBERT-PRF ReRanker, respectively, the MAP (y-axis) performance for different f_e (x-axis) selected from K clusters



(a) Cluster centroids, $K = 8$. (b) Cluster centroids, $K = 64$.

Figure 4.4: Embeddings selected using different number of clustering centroids K for the query ‘do goldfish grow’; point size is representative of the magnitude of IDF.

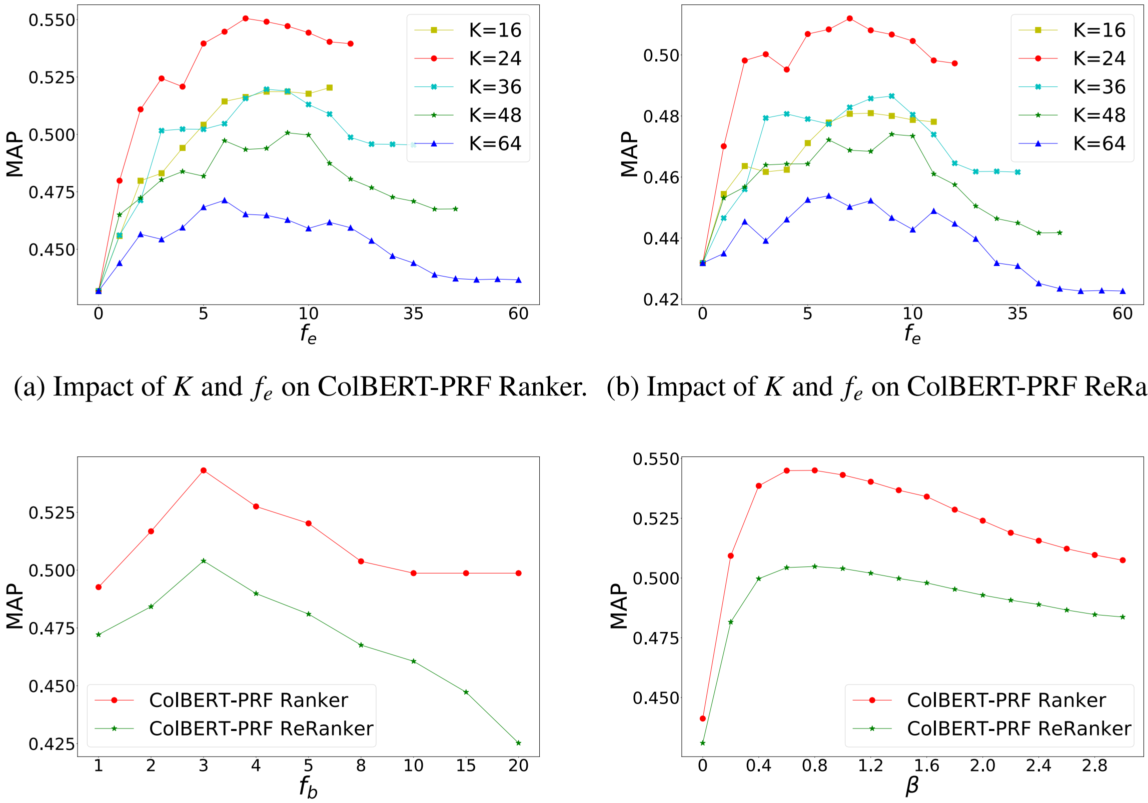
(different curves). We observe that, with the same number of clusters and expansion embeddings, ColBERT-PRF Ranker exhibits a higher MAP performance than ColBERT-PRF ReRanker – as we also observed in Section 4.3.3.1.

Then, for a given f_e value, Figures 4.5(a) and (b) show that the best performance is achieved by ColBERT-PRF when using $K = 24$. To explain this, we refer to Figure 4.4 together with Figure 4.2(b), which both show the centroid embeddings obtained using different numbers of clusters K. Indeed, if the number of clusters K is too small, the informativeness of the returned embeddings would be limited. For instance, in Figure 4.4(a), the centroid embeddings represent stopwords such as ‘in’ and ‘##′ ’ are included, which are unlikely to be helpful for retrieving more relevant passages. However, if K is too large, the returned embeddings contain more noise, and hence are not suitable for expansion – for instance, using $K = 64$, feedback embeddings representing ‘innocent’ and ‘stunt’ are identified in Figure 4.4(b), which could cause a topic drift.

Next, we analyse the impact of the number of feedback passages, f_b. Figure 4.5(c) reports the MAP performance in response to different number of f_b for both ColBERT-PRF Ranker and ReRanker. We observe that, when $f_b = 3$, both Ranker and ReRanker obtain their peak MAP values. In addition, for a given f_b value, the Ranker exhibits a higher performance than the ReRanker. Similar to existing PRF models, we also find that considering too many feedback passages causes a query drift, in this case by identifying unrelated embeddings.

Finally, we analyse the impact of the β parameter, which controls the emphasis of the expansion embeddings during the final passage scoring. Figure 4.5(d) reports MAP as β is varied for ColBERT-PRF Ranker and ReRanker. From the figure, we observe that in both scenarios, the highest MAP is obtained for $\beta \in [0.6, 0.8]$, but good effectiveness is maintained for higher values of β, which emphasises the high utility of the centroid embeddings for effective retrieval.

Overall, in response to RQ4.3, we find that ColBERT-PRF, similar to existing PRF approaches, is sensitive to the number of feedback passages and the number of expansion embeddings that are



(a) Impact of K and f_e on ColBERT-PRF Ranker. (b) Impact of K and f_e on ColBERT-PRF ReRanker.

(c) Impact of pseudo-relevance feedback size f_b. (d) Impact of expansion embedding weight β.

Figure 4.5: MAP on the TREC 2019 query set while varying the number of clusters (K), number of expansion embeddings (f_e), as well as the feedback set size f_b and expansion embedding weight β. $\beta = 0$ & $f_e = 0$ correspond to the original ColBERT.

added to the query (f_b & f_e) as well as their relative importance during scoring (cf. β). However, going further, the K parameter of KMeans has a notable impact on performance: if too high, noisy clusters can be obtained; too low and the obtained centroids can represent stopwords. Yet, the stable and effective results across the hyperparameters demonstrate the overall promise of ColBERT-PRF.

4.3.3.4 RQ4.4 - Semantic Matching by ColBERT-PRF

We now analyse the expansion embeddings and the retrieved passages in order to better understand the behaviour of ColBERT-PRF, and why it demonstrates advantages over traditional (sparse) QE techniques.

Firstly, it is useful to inspect tokens corresponding to the expansion embeddings. Table 4.3[10] lists three example queries from both the TREC 2019 and 2020 query sets and their tokenised forms as well as the expansion tokens generated by the ColBERT-PRF model. For a given query, we

[10] In Table 4.3, the expansion embedding '(breeds|##kshi)', which appears for each query, is projected to be close to the embedding of the [D] token, which ColBERT places in each passage.

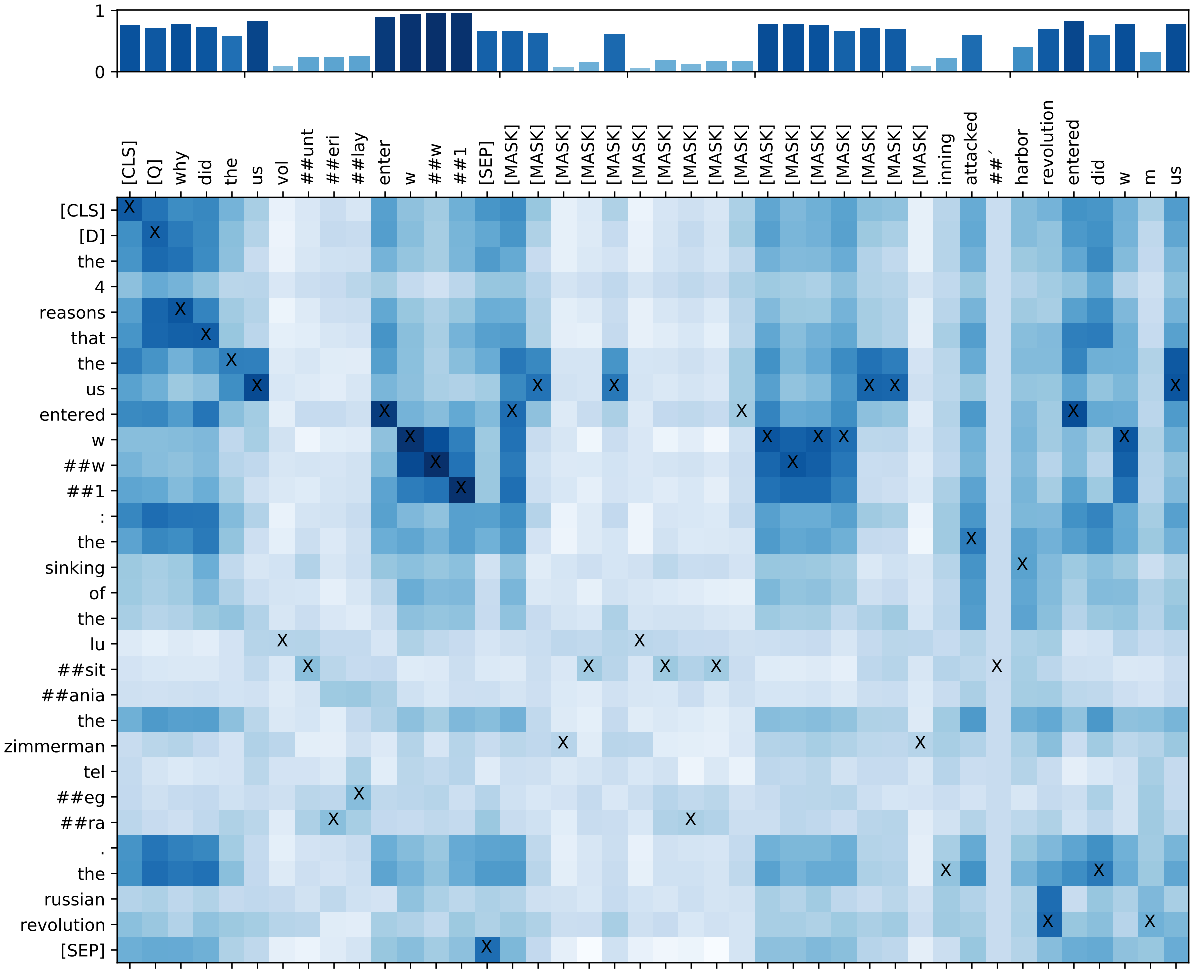


Figure 4.6: ColBERT-PRF interaction matrix between query (qid: 106375) and passage (docid: 4337532) embeddings. The darker shading indicate a higher similarity. The highest similarity among all the passage embeddings for a given query embedding is highlight with a X symbol. The histogram depicts the magnitude of contribution for each query embedding to the final score of the passage.

used our default setting for the ColBERT-PRF model, i.e., selecting ten expansion embeddings; Equation (4.2) is used to resolve embeddings to tokens. On examination of Table 4.3, it is clear to see the relation of the expansion embeddings to the original query - for instance, we observe that expansion embeddings for the tectonic concept of active margin relate to ‘oceanic’, ‘volcanoes’ and ‘continental’ ‘plate’. Overall, we find that most of the expansion tokens identified are credible supplementary information for each user query and can indeed clarify the information needs.

To answer RQ4.4, we further conduct analysis to measure the ability to perform semantic matching within the ColBERT MaxSim operation. Indeed, as the ColBERT model’s matching behaviour is performed using contextualised BERT embeddings for each token, polysemous words (which have the same surface form, but different meanings) will have distinct embeddings, while synonymous words will have similar embeddings. Hence it is possible to see the extent that synonymous words, or more generally, semantic matches – where the query token is not matched

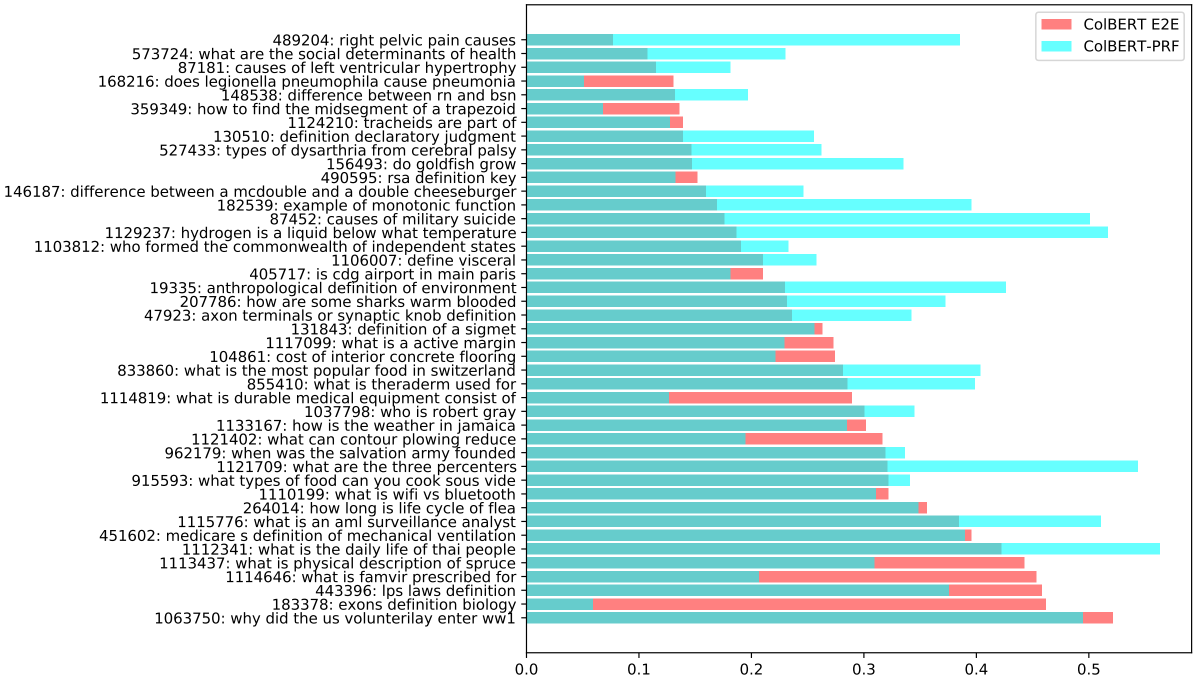
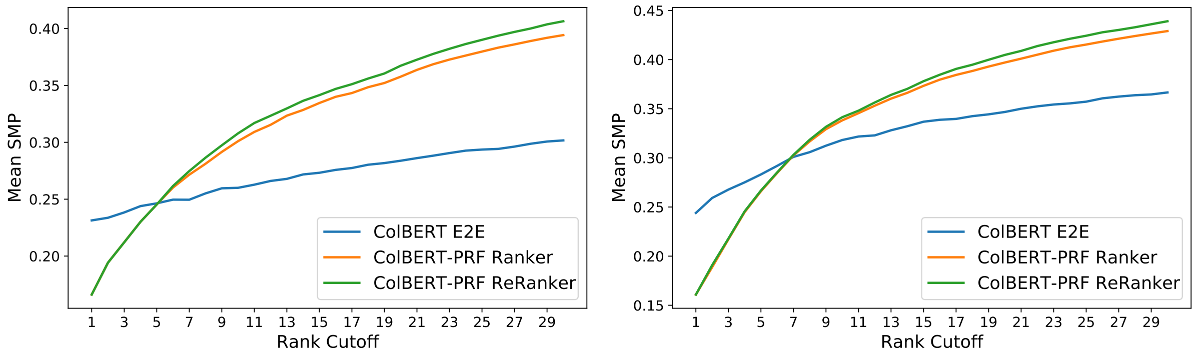


Figure 4.7: Per-query semantic matching proportion measurements (measured to rank 10) for the ColBERT E2E (shown as red bars) and ColBERT-PRF (shown as cyan bars) models on the TREC 2019 passage ranking query set.



(a) Mean SMP on TREC 2019

(b) Mean SMP on TREC 2020

Figure 4.8: Mean Semantic Matching Proportion (Mean SMP) as rank varies.

with an embedding representing the same word in the document – are occurring during retrieval. Indeed, we see this as a particular advantage for the ColBERT multiple representation model, which is not possible for models such as ANCE where the whole query and the whole document are each represented in a single representation embedding.

In particular, we examine which of the query embeddings match most strongly with a passage embedding that corresponds to exactly the same token - a so called *exact match*; in contrast a *semantic match* is a query embedding matching with a passage embedding which has a different token id. Indeed, in (Formal et al., 2021b), the authors concluded that ColBERT is able to conduct exact matches for important terms based on their embedded representations. In contrast, little

Table 4.3: Examples of the expanded queries by the ColBERT PRF model on the TREC 2019 & 2020 query sets. The symbol | denotes that there are multiple tokens that are highly likely for a particular expansion embedding. Token with darker red colour indicate its higher effectiveness contribution.

Original query terms	Original query tokens	Most likely tokens for expansion embeddings
TREC 2019 queries		
what is a active margin	what is a active margin	(by|opposition) oceanic volcanoes ##cton (margin|margins) (breeds|##kshi) continental plate an each
what is wifi vs bluetooth	what is wi ##fi vs blue ##tooth	##tooth (breeds|##kshi) phones devices wi ##fi blue systems access point
what is the most popular food in switzerland	what is the most popular food in switzerland	##hs (swiss|switzerland) (influences|includes) (breeds|##kshi) potato (dishes|food) (bologna|hog) cheese gr (italians|french)
TREC 2020 queries		
what is mamey	what is ma ##me ##y	(is|upset) (breeds|##kshi) flesh sap ##ote fruit ma ##me (larger|more) central
average annual income data analyst	average annual income data analyst	(analyst|analysts) (breeds|##kshi) (55|96) (grow|growth) salary computer tax 2015 depending ##k
do google docs auto save	do google doc ##s auto save	(breeds|##kshi) doc (to|automatically) google document save (saves|saved) drive (changes|revisions) (back|to)

work has considered the extent that ColBERT-based models perform semantic (i.e. non-exact) matching. Thus, firstly, following (Macdonald et al., 2021b), we look into the interaction matrix between the query and passage embeddings. Figure 4.6 describes the interaction matrix between the query "why did the us voluntarily enter ww1" expanded with 10 expansion embeddings and its top returned passage embeddings[11].

From Figure 4.6, we see that some query tokens, such as 'the', 'us', 'w' and '##w', experience exact matching as these tokens are in the same form with their corresponding returned highest MaxSim scored passage tokens. In contrast, the remaining query tokens are performing semantic matching to the passage as their corresponding passage tokens with the highest MaxSim score are in different lexical forms, for instance, query token 'why' matches with passage token 'reason'. In particular, the expansion token 'revolution' and 'entered', which does not exist in the original token but are expanded using ColBERT-PRF, also performs the exact matching. In addition, the expansion tokens such as 'attacked' and 'harbour' further perform semantic matching to the passages. This further indicates the usefulness of the expansion tokens to improve the matching performance between query and passage pairs.

[11] We use a FAISS index to map embeddings back to the most likely token.

To quantify the extent that semantic matching takes place, we define a new measure that inspects the MaxSim, and determines whether each query embedding is matched with the same token (exact match) vs. an inexact (semantic) match with a different token. Formally, let t_i and t_j respectively denote the token id of the i-th query embedding and j-th passage embedding, respectively. Given a query q and the set R_k of the top-ranked k passages, the *Semantic Match Proportion* (SMP) at rank cutoff k w.r.t. q and R_k is defined as:

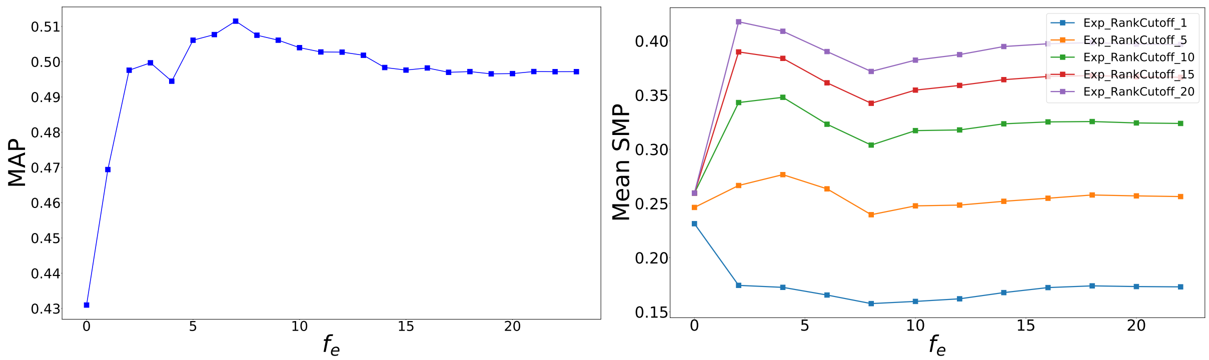
$$SMP(q,R_k) = \sum_{d \in R_k} \frac{\sum_{i \in \mathsf{toks}(q)} \mathbb{1}[t_i \neq t_j] \cdot \max_{j=1,\dots,|d|} \phi_{q_i}^T \phi_{d_j}}{\sum_{i \in \mathsf{toks}(q)} \max_{j=1,\dots,|d|} \phi_{q_i}^T \phi_{d_j}}, \tag{4.5}$$

where $\mathsf{toks}(q)$ returns the indices of the query embeddings that correspond to BERT tokens, i.e., not [CLS], [Q], or [MASK] tokens[12], R_k is the top ranked k passages, and $\mathbb{1}[]$ is the indicator function. The definition of exact vs. semantic matching is overall useful to compare the matching behaviour between different models. However, polysemous words may be counted as exact matches (e.g. ‘bank’ in ‘river bank’ vs. ‘national bank of greece’) while they may match semantically. Nevertheless, the embedding similarity for such occurrences of ‘bank’ would be low. As a consequence, we argue that the contribution of the polysemous words to the lexical category of SMP, when calculated on top-ranked documents, is likely to be negligible.

Figure 4.7 depicts the per-query semantic matching proportion calculated at the rank cutoff 10 for the ColBERT-PRF and ColBERT E2E models on the TREC 2019 query set. From the figure, we observe that when the expansion embeddings are added to the original query by ColBERT-PRF, SMP is increased for most of the queries over the original ColBERT E2E model. Next, on both TREC 2019 and TREC 2020 query sets, we investigate the impact of the rank cutoff k to the semantic match proportion on ColBERT-PRF model instantiated as Ranker and ReRanker models as well as the ColBERT E2E model, which is portrayed in Figure 4.8. In general, from Figure 4.8, we can see that Mean SMP grows as the rank cutoff k increases - this is expected, as we know that ColBERT prefers exact matches, and the number of exact matches will be decreased by rank (resulting in increasing SMP). However, ColBERT-PRF (both Ranker and Reranker) have, in general, higher SMP than the original ColBERT ranking. This verifies the results from Figure 4.7. The interesting exception is at the very highest ranks, where both ColBERT-PRF approaches exhibit lower SMP than the baseline. This suggests that at the very top ranks, ColBERT-PRF exhibits a higher preference for exact token matches than the E2E baseline. However, overall, the higher SMPs exhibited by ColBERT-PRF indicate that, at deeper ranks, the embedding-based query expansion has the ability to retrieve passages with a less lexical exact match between the query and passage embeddings.

In addition, we further investigate the potential for topic drift when applying ColBERT-PRF with the different number of expansion embeddings on the TREC 2019 queries. In particular, in

[12] Indeed, [CLS], [Q], and [MASK] do not correspond to actual WordPiece tokens originating from the user’s query and hence can never have exact matches, so we exclude them from this calculation.



(a) Retrieval effectiveness (MAP) in terms of different number of expansion embeddings f_e on ColBERT-PRF ReRanker.

(b) Mean SMP performance in terms of different number of expansion embeddings f_e on ColBERT-PRF ReRanker.

Figure 4.9: Potential topic drift analysis for ColBERT-PRF ReRanker on the TREC 2019 query set.

Figure 4.9a[13] we measure retrieval effectiveness (MAP) as the number of expansion embeddings is varied and, in Figure 4.9b, we present Mean SMP (y-axis) calculated upon the retrieved results after PRF, at different rank cutoffs (curves), also as the number of expansion embeddings is varied (x-axis).

From Figure 4.9a, we can see that $f_e = 8$ gives the highest (MAP) effectiveness (as also shown earlier in Figure 4.5b). At the same time, from Figure 4.9b, we observe that (1) for $2 \leq f_e \leq 8$, Mean SMP falls; (2) however, for $f_e > 8$, Mean SMP rises again. This trend is apparent when Mean SMP is analysed for 5 or more retrieved passages. This suggests that with more than 8 expansion embeddings selected, excessive semantic matching occurs (Figure 4.9b) and effectiveness approaches MAP 0.50 (Figure 4.9a). As expansion embeddings are selected by using the IDF of the corresponding token, this suggests that given the size of the feedback set (3 passages, with length upto 180 tokens and on average 77 tokens), for more than 8 embeddings we are starting to select non-informative expansion embeddings that can only be semantically matched in the retrieved passages, and hence there is no further positive benefit in terms of effectiveness. However, as effectiveness does not markedly decrease for $f_e > 8$, this indicates that there is little risk of topic drift with ColBERT-PRF, due to the contextualised nature of the expansion embeddings. Overall, these analyses answer RQ4.4.

4.4 Document Ranking Effectiveness of ColBERT-PRF

After assessing the effectiveness of ColBERT-PRF on passage ranking for the in-domain evaluation in the previous section, we further demonstrate the performance of ColBERT-PRF on document ranking. In this task, documents are longer than passages, hence they need to be divided into smaller chunks, with lengths comparable to those of passages. Moreover, in docu-

[13] This is a subset of the curves presented earlier in Figure 4.5b, repeated here for ease of reference.

ment ranking we do not fine-tune the ColBERT model on the new collection due to the limited number of queries available; hence, we leverage the ColBERT model trained on the MSMARCO as detailed in Section 4.3.2, e.g., in a zero shot out-of-domain evaluation setting. Thus, in this section, we focus on testing the effectiveness of our proposed ColBERT-PRF for the MSMARCO document retrieval task and the TREC Robust04 document retrieval task. Research questions and experimental setup for document ranking experiments are detailed in Section 4.4.1 and Section 4.4.2, respectively. Results and analysis are discussed in Section 4.4.3.

4.4.1 Research Questions

Our document ranking experiments address the following research questions:

- RQ4.5: Can our pseudo-relevance feedback mechanism enhance the retrieval effectiveness of dense retrieval models, i.e., ColBERT-PRF model outperform ColBERT, ANCE and ANCE-PRF dense retrieval models for document retrieval task?
- RQ4.6: How does ColBERT-PRF compare to other existing baseline and state-of-the-art approaches for document retrieval task, namely:

    (a) lexical (sparse) baselines, including using PRF,

    (b) BERT-QE Reranking models,

    (c) embedding-based query expansion models, namely the three variants of the CEQE model: CEQE-Max, CEQE-Centroid and CEQE-Mul?

4.4.2 Experimental Setup

In this section, we detail the experimental setup for ColBERT-PRF on the document ranking task, including the dataset and metrics used, the implementation detail as well as the baselines.

4.4.2.1 Dataset & Measures

In this section, we evaluate our ColBERT-PRF on document ranking task using MSMARCO document and Robust04 document datasets (cf. Section 2.5.1). To test the retrieval effectiveness of the ColBERT-PRF model, we use the 43 test queries from the TREC Deep Learning Track 2019 and 45 test queries from the TREC Deep Learning Track 2020, respectively. In addition, we also conducted the evaluation using 250 title-only and description-only query sets from the TREC Robust04 document ranking task.

We report the following metrics for MSMARCO document ranking tasks, namely the normalised discounted cumulative gain (NDCG) calculated at rank 10, Mean Average Precision (MAP) at rank 1000 as well as Recall calculated at ranks 100 and 1000. For the Robust04 experiments, we use the same metrics used for passage ranking tasks in Section 4.3.2. For significance testing, we use the paired t-test ($p < 0.05$) and apply the Holm-Bonferroni multiple testing correction.

4.4.2.2 Implementation and Settings

As the length of documents in these corpora are too long to be fitted into the BERT model, and in particular our trained ColBERT model[14] (limited to 512 and 180 BERT WordPiece tokens, respectively), we split long documents into smaller passages and index the generated passages following (Dai and Callan, 2019a). In particular, when building the index for each document corpora, a sliding window of 150 tokens with a stride of 75 tokens is applied to split the documents into passages. All the passages are encoded into a FAISS index. At retrieval time, FAISS retrieves $k' = 1000$ document embeddings for every query embedding. The final score for each document is obtained by taking its highest ranking passage, a.k.a., it's *max passage*.
To ensure a fair comparison, we apply passaging for all other indices used in this section, including the Terrier inverted index, i.e., the ANCE dense index[15]. Similarly, all PRF methods are applied on feedback *passages*, and max passage applied on the final ranking of passages.
Finally, we follow the same ColBERT-PRF implementation as introduced in Section 4.3.2. For query expansion settings, we follow the default settings for the passage ranking task in Section 4.3.3, which is 10 expansion terms obtained from 3 feedback passages[16] and $K = 24$ clusters.

4.4.2.3 Baselines

To test the effectiveness of our ColBERT-PRF model on the document ranking task, we compare with all the baseline models we used for the passage ranking task except the *Neural Augmentation Approaches*, due to the high GPU indexing time required for performing the doc2query and DeepCT processing for these large document corpora.

4.4.3 Document Ranking Results

In this section, we further investigate the effectiveness of our proposed ColBERT-PRF for document ranking tasks. Table 4.4 and Table 4.5 present the performance of ColBERT-PRF models as well as the baselines on the MSMARCO document dataset and the Robust04 dataset, respectively.

[14] It is a common practice to use models trained on the MSMARCO passage corpus (Nguyen et al., 2016) for document retrieval (e.g. (Li et al., 2020, Nogueira et al., 2020)). [15] While this is necessary for a fair comparison, it results in a small degradation in effectiveness for the sparse baselines - this has also been observed by the authors of Anserini - see `github.com/castorini/anserini/blob/master/src/main/python/passage_retrieval/example/`

[16] We also tried filtering passages from the same document before applying PRF. We observed no significant improvements across multiple measures.

Table 4.4: the MSMARCO Document corpus. Comparison with baselines. Superscripts a...p denote significant improvements over the indicated baseline model(s). The highest value in each column is boldfaced.

	TREC 2019 (43 queries)				TREC 2020 (45 queries)			
	MAP	nDCG@10	Recall@100	Recall@1000	MAP	nDCG@10	Recall@100	Recall@1000
Lexical Retrieval Approaches								
BM25 (a)	0.3145	0.5048	0.3891	0.6975	0.3650	0.4709	0.6095	0.8143
BM25+BERT (b)	0.3797	0.6279	0.4363	0.6977	0.4387	0.5993	0.6646	0.8147
BM25+ColBERT (c)	0.3862	0.6503	0.4378	0.6970	0.4390	0.6144	0.6580	0.8155
BM25+RM3 (d)	0.3650	0.5411	0.4203	0.7304	0.3822	0.4770	0.6380	0.8311
BM25+RM3+BERT (e)	0.3973	0.6330	0.4466	0.7304	0.4470	0.5981	0.6646	0.8305
BM25+RM3+ColBERT (f)	0.4083	0.6633	0.4506	0.7300	0.4467	0.6074	0.6580	0.8305
Dense Retrieval Models								
ANCE (g)	0.2708	0.6468	0.3443	0.5349	0.4050	0.6256	0.5682	0.7197
ColBERT E2E (h)	0.3195	0.6342	0.3880	0.5642	0.4290	0.6113	0.6351	0.7951
BERT-QE Reranking Models								
BM25 + RM3 + ColBERT + BERT-QE (i)	**0.4340**	**0.6850**	**0.4626**	0.7298	0.4728	**0.6268**	0.6848	0.8310
ColBERT E2E + BERT-QE (j)	0.3358	0.6668	0.3953	0.5642	0.4478	0.6244	**0.7141**	0.7951
Embedding based Query Expansion								
CEQE-Max (k)	0.3778	0.5176	0.4313	**0.7462**	0.3956	0.4729	0.6546	**0.8410**
CEQE-Centroid (l)	0.3765	0.5103	0.4312	0.7432	0.3968	0.4746	0.6540	0.8390
CEQE-Mul (m)	0.3680	0.4959	0.4207	0.7360	0.3937	0.4809	0.6467	0.8351
ColBERT-PRF Models								
ColBERT-PRF Ranker (β=1)	0.3851^{ghj}	0.6681^{adklm}	0.4467^{aghj}	0.6252^{g}	**0.4885**^{adghklm}	0.6146^{adklm}	0.7120^{acdfghm}	0.8128^{g}
ColBERT-PRF ReRanker (β=1)	0.3473^{gh}	0.6688^{adklm}	0.4283^{ghj}	0.5459	0.4739^{adgklm}	0.6171^{adklm}	0.6933^{agh}	0.7782^{g}

4.4.3.1 RQ4.5 - Effectiveness of ColBERT-PRF for Document Ranking

Similar to the passage retrieval task, in this section we validate the effectiveness of the pseudo-relevance feedback technique for the ColBERT dense retrieval model on the document retrieval task. On analysing Table 4.4, we found that both ColBERT-PRF Ranker and ReRanker models significantly outperform both the single representation dense retrieval, namely ANCE, and the multiple representation dense retrieval model, namely ColBERT E2E, in terms of MAP and Recall on both TREC 2019 and TREC 2020 query sets. In particular, the application of ColBERT-PRF leads to upto 21% and 14% improvements over ColBERT E2E in terms of MAP for TREC 2019 and TREC 2020 query sets, respectively.

Indeed, ColBERT-PRF outperforms all document retrieval runs to the TREC 2019 Deep Learning track, exceeding the highest observed MAP by 23% in terms of MAP. Similarly, on the TREC 2020 query set, the MAP observed is markedly above that attained by the second-ranked group on the leaderboard (Craswell et al., 2021b).[17] In terms of nDCG@10, ColBERT-PRF outperforms both the ANCE and ColBERT E2E models on both MSMARCO query sets. Moreover, both the ColBERT-PRF Ranker and ReRanker models significantly outperform the ColBERT and ANCE models w.r.t. Recall@100, indicating the effectiveness of the ColBERT-PRF refined query representations.

Similarly, when comparing the performances of ColBERT-PRF with the dense retrieval models without pseudo-relevance feedback on Robust04 in Table 4.5, we note that both ColBERT-PRF Ranker and ReRanker models are markedly improved over the ANCE and ColBERT E2E models on MAP, nDCG@10 and Recall on both title-only and description-only type of queries. Overall,

[17] The first ranked group used expensive document expansion techniques.

Table 4.5: the Robust corpus. Comparison with baselines. Superscripts a...p denote significant improvements over the indicated baseline model(s). The highest value in each column is boldfaced.

	Robust title (250 queries)				Robust description (250 queries)			
	MAP	nDCG@10	MRR@10	Recall	MAP	nDCG@10	MRR@10	Recall
Lexical Retrieval Approaches								
BM25 (a)	0.2319	0.4163	0.6330	0.6758	0.2193	0.3966	0.6570	0.6584
BM25+BERT (b)	0.2550	0.4820	0.7290	0.6819	0.2723	0.4709	0.7293	0.6721
BM25+ColBERT (c)	0.2770	0.4753	0.7307	0.6821	0.2658	0.4728	0.7349	0.6684
BM25+RM3 (d)	0.2542	0.4244	0.6139	0.7007	0.2619	0.4182	0.6277	0.7008
BM25+RM3+BERT (e)	**0.2884**	**0.4839**	**0.7343**	0.7037	0.2814	0.4708	0.7251	0.7081
BM25+RM3+ColBERT (f)	0.2840	0.4758	0.7277	0.7058	0.2766	0.4739	0.7419	0.7068
Dense Retrieval Models								
ANCE (g)	0.1605	0.3713	0.6096	0.5410	0.1919	0.4242	0.7002	0.5794
ColBERT E2E (h)	0.2327	0.4446	0.7011	0.6076	0.2175	0.4352	0.6853	0.6054
BERT-QE Reranking Models								
BM25 + RM3 + ColBERT + BERT-QE (i)	0.2762	0.4407	0.6302	0.7072	**0.2926**	**0.4857**	**0.7369**	0.7076
ColBERT E2E + BERT-QE (j)	0.2395	0.4523	0.6973	0.6078	0.2289	0.4468	0.6904	0.6055
Embedding based Query Expansion								
CEQE-Max (l)	0.2829	0.4318	0.6334	**0.7494**	0.2745	0.4224	0.6461	0.7232
CEQE-Centroid (m)	0.2818	0.4299	0.6305	0.7457	0.2746	0.4217	0.6475	**0.7278**
CEQE-Mul (n)	0.2764	0.4267	0.6225	0.7375	0.2672	0.4076	0.6146	0.7256
ColBERT-PRF Models								
ColBERT-PRF Ranker (β=1)	0.2715^{adghj}	0.4670^{adgh}	0.6836^{dglmn}	0.6476^{ghj}	0.2627^{aghj}	0.4605^{ah}	0.6678	0.6347^{ghj}
ColBERT-PRF ReRanker (β=1)	0.2642^{adghj}	0.4682^{adgh}	0.6837^{dg}	0.6158^{g}	0.2592^{aghj}	0.4624^{ah}	0.6681	0.6289^{ghj}

between the Ranker and ReRanker ColBERT-PRF models, we find that ColBERT-PRF Ranker is more effective than ColBERT-PRF ReRanker, likely due to its increased Recall, consistent with those obtained from the passage ranking task (Section 4.3). Thus, in response to RQ4.5, we conclude that our ColBERT-PRF is effective at improving ColBERT E2E on document ranking tasks, similar to the improvements observed in Section 4.3.

4.4.3.2 RQ4.6 - Comparison to Baselines

In the following, we compare the effectiveness of the ColBERT-PRF model with various baselines. From Table 4.4, we find that ColBERT-PRF instantiated as the Ranker model significantly improves over the BM25-based lexical retrieval baselines and the ColBERT E2E with BERT-QE as the reranker, as well as all the CEQE variants models in terms of the nDCG@10 and Recall@100 metrics on the TREC 2019 query set. In addition, for the TREC 2020 query set, ColBERT-PRF significantly improves over all the baselines except those with BERT-based neural reranking models, namely BERT, ColBERT and BERT-QE, in terms of the MAP and Recall@100 metrics.

Now let's analyse the performance of ColBERT-PRF models on Robust04 query sets. From Table 4.5, we observe that ColBERT-PRF models significantly outperform the BM25 on both query sets and markedly outperform over BM25 + RM3 on title-only queries. In addition, ColBERT-PRF shows a similar performance with CEQE models in terms of MAP but exhibits marked improvements in terms of nDCG@10 and MRR@10. Moreover, when comparing with the models with neural rerankers, both ColBERT-PRF Ranker and ReRanker models significantly outperform the ColBERT E2E + BERT-QE baseline and exhibit comparable performance to the other neural reranker models. However, we argue that the limited performance of ColBERT-PRF compared with the BERT-based reranking models for the Robust04 query sets comes from the two following aspects: firstly, we used a zero-shot setting of ColBERT model for the document ranking tasks, in that the ColBERT model was not trained on the larger document datasets; second, we didn't perform further parameter tuning for ColBERT-PRF on the document ranking task. Thus, in response to RQ4.6, we find that ColBERT-PRF is more effective than most of the baseline models and comparable to the BERT-based neural reranking models.

4.5 Measuring the Informativeness of Expansion Embeddings of ColBERT-PRF

In this section, we investigate the effectiveness of the three variants of the ColBERT-PRF model using different techniques to measure the informativeness of the expansion embeddings. The strategies are detailed in Section 4.5.1. Accordingly, a research question is posed in Section 4.5.2, with a corresponding experimental setup. Finally, Section 4.5.3 presents the performance and analysis of the three ColBERT-PRF variants.

4.5.1 Methodology

In Section 4.2.2 we proposed to map each expansion embedding back to its most likely token, and use the IDF of that token to measure the importance σ of each expansion embedding υ_i generated by ColBERT-PRF. This results in a weight, $\sigma(\upsilon_i)$, that is used in the expanded MaxSim calculation (Equation (4.4)). Indeed, notions of document frequency or collection frequency are commonly used in PRF models to measure expansion terms (Amati, 2003). The intuition behind this is that if a term appears more frequently in the feedback documents than in the whole corpus, the term is taken as an informative term. In contrast, terms that occur frequently in the corpus will not discriminate well relevant documents from other documents in the collection. In this section, we revisit the use of IDF in ColBERT-PRF, by additionally using the collection frequency of the token, while also examining the corresponding embeddings of the tokens. Indeed, in addition to the document frequency focus of IDF, the collection frequency is also useful to reflect the informativeness of a term within the whole collection, measured as follows:

$$\sigma_{ICTF}(t) = \log\left(\frac{|D|+1}{tf(t,D)+1}\right), \tag{4.6}$$

where $|D|$ is the number of terms in the collection D and $tf(t,D)$ is the number of occurrences of expansion term t in the whole collection D.

However using either IDF or ICTF as expansion embedding weights does not consider the contextualised nature of the embeddings - that different tokens can have distinct meanings, and these may be more or less useful for retrieval. The use of IDF or ICTF can mask such distinctions. Hence, we examine a further method based directly on the embedded representations. In particular, for each token, we examine all corresponding embeddings in the index, and determine how ‘focused’ these are - we postulate that a token with more focused embeddings will only have a single meaning (and therefore less polysemous), and hence more likely to be a good expansion embedding. Specifically, we measure the Mean Cosine similarity (MCos) for the embeddings of each token compared to the mean of all those embeddings:

$$\sigma_{MCos}(t) = \frac{1}{tf(t,D)} \sum_{j=1}^{tf(t,D)} \cos(\Upsilon, \phi_{c_j}), \tag{4.7}$$

where Υ is the element-wise average embedding of all embeddings in the index for token t. MCos is intended to approximate the semantic coherence of the embeddings for a given token. The expansion embeddings of more coherent tokens are given a higher weight in ColBERT-PRF.

4.5.2 Research Question & Experimental Setup

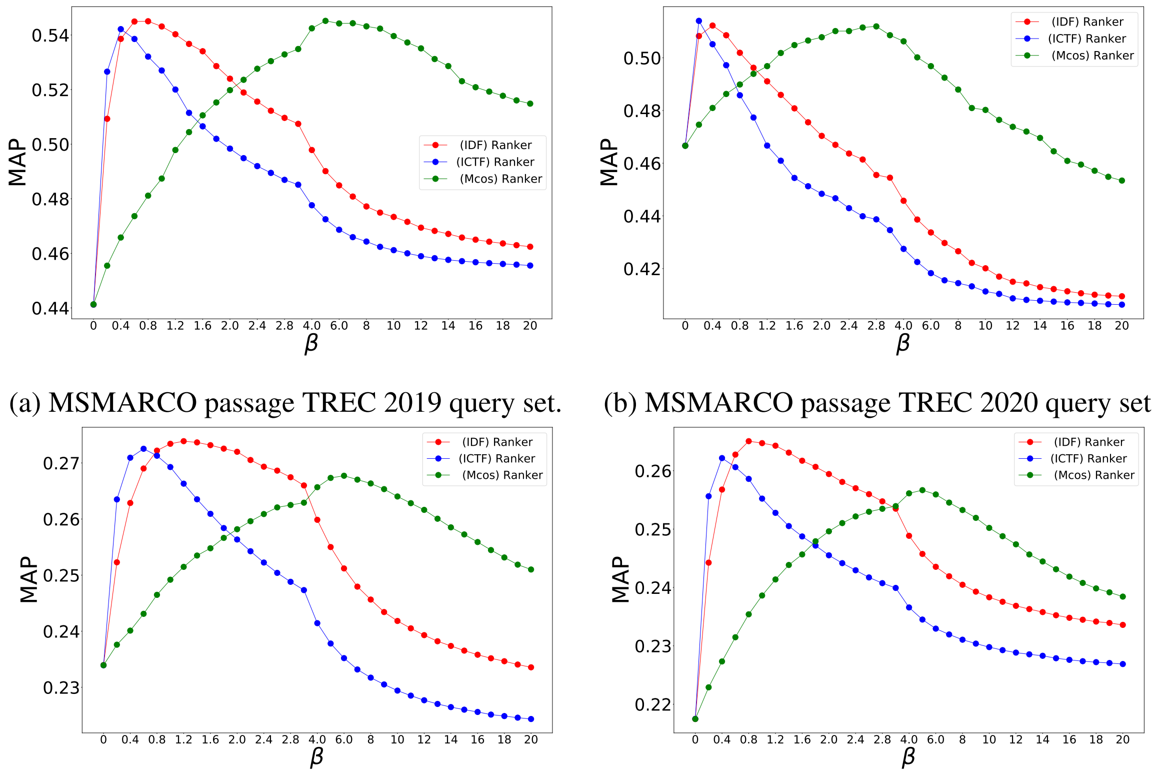
Our informativeness measurement experiments address the following research question:

- RQ4.7: What is the impact of the effectiveness ColBERT-PRF using different informativeness of expansion embedding measurements methods, namely the IDF weighting method, ICTF weighting method and the MCos weighting method?

In our experiments addressing RQ4.7, while testing IDF, ICTF and MCos importance measures, we vary the parameter of ColBERT-PRF that controls the overall weight of the expansion embeddings, β. We do not normalise the various importance measures $\sigma_{IDF}(t)$, $\sigma_{ICTF}(t)$ and $\sigma_{MCos}(t)$ – their inherent differences in scales are addressed by varying β.
Dataset: The query sets we used to demonstrate the effectiveness of the three variants of ColBERT-PRF proposed are the MSMARCO passage TREC 2019 and TREC 2020 passage query sets for passage retrieval task and the Robust title and description query sets for document retrieval task.
Measures: Mean Average Precision (MAP) is used as the main metric.

4.5.3 Results

Figure 4.10 shows the impacts of the retrieval effectiveness of the different weighting methods while β is varied, in terms of MAP, for ColBERT-PRF for both the MSMARCO passage ranking task and the Robust04 document ranking task. Specifically, for the passage ranking task, we measure the retrieval effectiveness on both the TREC 2019 and TREC 2020 passage ranking queries, and using title-only and description-only types of queries of Robust04.



(a) MSMARCO passage TREC 2019 query set.

(b) MSMARCO passage TREC 2020 query set

(c) Robust04 title query set.

(d) Robust04 desciption query set.

Figure 4.10: Influence of different weighting methods. $\beta = 0$ corresponds to the original ColBERT.

On analysing the figure, we see that, for both of the TREC 2019 and TREC 2020 query sets, the peak MAP scores for all the three weighting methods are the same, approximately with MAP=0.54 and MAP=0.51 respectively. In addition, according to Figure 4.10a and 4.10b, the overall trend for IDF and ICTF weighting methods are the same and both reach the highest MAP score with $\beta \in [0.4, 0.8]$. When we compare with IDF and ICTF, we see that MCos with $\beta \in [4.0, 6.0]$ exhibits the highest MAP performance. These trends allow us to draw the following observations: the lines for IDF and ICTF are very similar, varying only in terms of the β value needed to obtain the highest MAP; In contrast, the MCos weighting method achieves a similar maximum MAP, but at a larger β value – this is due to the lack of common normalisation. Indeed, as the maximum MAP values obtained are similar for IDF, ICTF and MCos, this suggests that the MCos is correlated with IDF, and that the statistical approaches are sufficient for measuring expansion embedding importance. A closer analysis of IDF and ICTF, as calculated on the BERT tokens, found that they exhibit a very high correlation (Spearman's ρ of $\sim$1.00 on the MSMARCO passage corpus). This is indeed higher than the correlation observed on a traditional sparse Terrier inverted index (which uses a more conventional tokeniser) of 0.95 on the MSMARCO document index. The differences in correlations can be explained as follows: firstly, due to the use of WordPieces by the BERT tokeniser, which reduces the presence of long-tail tokens (which are tokenised to smaller WordPieces); secondly, passage corpora use smaller indexing units than document corpora, so it is less likely for terms to occur multiple times – this results in collection frequency being more correlated to document frequency.

For the Robust04 query set (Figure 4.10c and 4.10d), we see that while the peak MAP values for IDF and ICTF are again similar, the MCos weighting method gives lower MAP scores on the Robust04 title and description query sets. This suggests that using the coherence of a token's embeddings may not well indicate the utility of the expansion embedding. Indeed, some tokens with high embedding coherence could be stopword-like in nature. This motivates the continued use of IDF and ICTF for identifying important expansion embeddings.

Overall, to address RQ4.7, we find that the statistical information, based on the IDF and ICTF weighting methods, is more stable than the MCos weighting method for different retrieval tasks. Use of IDF and ICTF were shown to be equivalent, due to the higher correlation between document frequency and collection frequency on passage corpora.

4.6 Efficient Variants of ColBERT-PRF

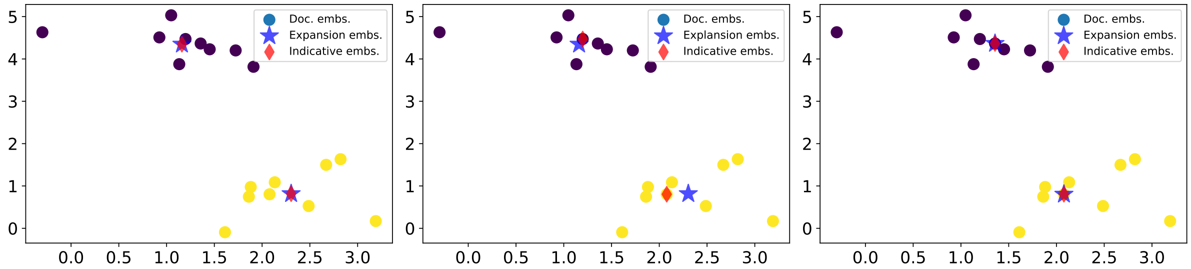
In Section 4.3.3, we noted the high mean response time of the ColBERT PRF approach. Higher response times are a feature of many PRF approaches, due to the need to analyse the contents of the feedback documents, and decide upon the expansion terms/embeddings. In this section, we investigate several efficient variants of our ColBERT-PRF model, by experimenting with different clustering approaches, as well as different retrieval configurations of ColBERT.

In particular, we describe different variants in Section 4.6.1. Two research questions and the implementation setup are detailed in Section 4.6.2.1. Results and analysis are discussed in Section 4.6.3.

4.6.1 ColBERT-PRF variants

The overall workflow of a ColBERT-PRF Ranker model can be described into five stages, as shown in Figure 4.1. These stages can be summarised as follows (for the ColBERT-PRF ReRanker model, the fourth stage ANN retrieval is omitted):

- Stage 1: First-pass FAISS ANN Retrieval
- Stage 2: First-pass exact ColBERT MaxSim re-ranking
- Stage 3: Clustering of Feedback Documents and Expansion Embedding Weighting
- Stage 4: Second-pass FAISS ANN Retrieval
- Stage 5: Second-pass exact ColBERT MaxSim re-ranking



(a) KMeans clustering.

(b) KMeans-Closest clustering.

(c) KMedoid clustering.

Figure 4.11: The illustration of different clustering methods. Dots in different colours indicate the document embeddings belonging to different clusters. Blue stars represents the expansion embedding, while the red diamond represents the indicative embedding used to measure the informativeness of the expansion embeddings.

In the following, we discuss changes to the clustering (Stage 3 above, Section 4.6.1.1) and ANN retrieval (Stages 1 & 4, Section 4.6.1.2).

4.6.1.1 Clustering

The default clustering technique in Stage 3 is the KMeans clustering algorithm. KMeans clustering is a widely used clustering method, which groups the samples into k clusters according to their Euclidean distance to each other. Hence, in ColBERT-PRF, given a set of document embeddings and the number of clusters expected to be returned, KMeans clustering is employed to return a list of representative centroid embeddings. Figure 4.11a provides an illustration of the KMeans clustering method. Indeed, as shown in Figure 4.11a, we notice that both cluster centroids (which can be applied as expansion embeddings for PRF) are distinct from the input embeddings. As a consequence, while measuring the importance and selecting the most informative ones among the representative centroid embeddings using IDF (or ICTF or MCos), we are required to map each centroid embedding to a corresponding token id. As the representative centroid embedding, by definition, is not an actual document embedding, we turn to the FAISS ANN index and apply Equation (4.2) to obtain a list of token ids (see Section 4.2.2). However, the main drawback of the above KMeans clustering method in ColBERT-PRF is that the procedure of looking up the most likely token for each of the K centroid embeddings requires another K FAISS lookups. To address this issue, we propose variants that avoid these additional FAISS lookups, by using the most likely token within each cluster - to do so, we recognise that the expansion embedding (which is added to the query) needs not perfectly alignment with the embedding used to measure informativeness, which we call the *indicative embedding*.

Our first proposed alternative strategy is called KMeans-Closest, which is still based on KMeans clustering but does not rely on additional FAISS lookups to obtain the most likely tokens. Once the K centroid embeddings are computed, for each centroid we identify the closest feedback document embedding in the corresponding cluster – the indicative embedding for each cluster – and we use its token id to measure the importance score, such as IDF of the expansion embeddings. As shown in Figure 4.11b, the indicative embeddings (the diamonds) are the closest actual document embeddings to the KMeans centroid embeddings (the blue stars).

Our second proposed clustering strategy is KMedoids (Khennak et al., 2019). The KMedoids algorithm returns the *medoid* of each cluster – the medoid is the most centrally located embedding of the input document embeddings. Thus, after applying clustering upon the feedback document embeddings, for each cluster, we obtain the medoid (an indicative embedding for the cluster) that is also an actual document embedding, and hence can be mapped back to a token id, without requiring additional FAISS lookups for each centroid. Figure 4.11c depicts both the expansion embeddings and the indicative embeddings are the returned medoid embeddings of the KMedoids clustering algorithm.

Overall, while the use of the KMeans-Closest and KMedoids methods can speed up the third stage of ColBERT-PRF, there might exist some potential risks (e.g., token id mismatch) thus hindering the effectiveness – hence, we report effectiveness as well as efficiency in our experiments.

4.6.1.2 ANN Retrieval

The overall ColBERT-PRF Ranker process encapsulates a total of 5 stages, as shown in Figure 4.1. An ANN retrieval stage is used in both stages 1 & 4, and hence forms a significant part of the workflow. Indeed, as highlighted in Section 4.1, for each given query embedding, the approximate nearest neighbour search produces k' document embeddings for each query embedding, which are then mapped to the corresponding documents, thereby forming an unordered *set* of candidate documents. However, the contribution of the different query embeddings to the final score of the document varies (cf. the contribution histogram in Figure 4.6)[18]. Therefore, it is not efficient to take upto $k' = 1000$ documents for each query embedding forward to the 2nd stage for accurate MaxSim scoring, as not all of these documents will likely receive high scores.

To this end, we experiment with using *Approximate Scoring* (Macdonald and Tonellotto, 2021) at the first stage, as well as in the later stage 4 retrieval. In particular, this approach makes use of the MaxSim operator applied on the *approximate* cosine scores of the ANN algorithm, to generate a *ranking* of candidates from the first stage. Indeed, as this is a ranking, rather than a set, then the number of the candidates k can be directly controlled, rather than indirectly through k'. While requires more computation in stage 1 (and has a small negative impact on the response time of that stage), it has marked overall efficiency benefits (Macdonald and Tonellotto, 2021) for ColBERT dense retrieval, as a smaller number of candidates can be passed to MaxSim without loss of recall.

More specifically, for the ColBERT-PRF instantiated as Ranker model, we apply the Approximate

[18] Indeed, in separate but orthogonal work (Tonellotto and Macdonald, 2021b), we show that query embeddings vary in their ability to recall relevant documents, and some can even be discarded (*pruned*) from the ANN search phase without significant loss of effectiveness.

Scoring technique only in the first stage or in both the first and fourth stages of the ColBERT-PRF-Ranker model. Indeed, as we only require the most relevant three feedback passages for effective PRF, accurately scoring thousands of passages retrieved by the 1st ANN stage is superfluous. For the ColBERT-PRF instantiated as the ReRanker model, we apply Approximate Scoring in the first stage. In addition, we further investigate the efficiency and effectiveness trade-off when implementing the different clustering techniques and the Approximate Scoring technique in the various ColBERT stages.

4.6.2 Experimental Setup

Now we describe the experimental setup for the efficient ColBERT-PRF variants, where Section 4.6.2.1 introduces the research questions and Section 4.6.2.2 introduces the dataset and measures. In addition, the experimental settings are detailed in Section 4.6.2.3.

4.6.2.1 Research Question & Experimental Setup

For the efficient ColBERT-PRF variants, we pose the following research questions:

- RQ4.8: What is the impact on efficiency and effectiveness of the ColBERT-PRF model using different clustering methods, namely the KMeans and KMeans-Closest clustering methods and the KMedoids clustering method?

- RQ4.9: What is the impact on efficiency and effectiveness of the ColBERT-PRF model when instantiated using Approximate Scoring?

4.6.2.2 Dataset & Measures

We compare the efficiency and the effectiveness of ColBERT-PRF model efficient variants on TREC 2019 and TREC 2020 query sets from MSMARCO passage.
For measuring the performance in terms of efficiency, we report the Mean Response Time (MRT) for each stage of the ColBERT-PRF model (described in Figure 4.1) and its overall MRT. Mean response times are measured with one Nvidia Titan RTX GPU (using a single thread for retrieval). In addition, we report the effectiveness performance with the metrics used in Section 4.3.2, namely MAP, nDCG@10, MRR and Recall. For significance testing, we use the paired t-test ($p < 0.05$) and apply the Holm-Bonferroni multiple-testing correction technique.

4.6.2.3 Experimental Setting

For both KMeans-Closest and KMedoids clustering, we reuse the default setting of the KMeans clustering algorithm, i.e., the number of clusters $K = 24$, the number of feedback documents $f_b = 3$, and the number of expansion embeddings $f_e = 10$. As for β, based on the conclusions obtained from Section 4.3, we pick the appropriate β for each query set, namely $\beta = 1$ and

$\beta = 0.5$ for the TREC 2019 and TREC 2020 passage ranking query sets, respectively. For the Approximate Scoring experiments, let k_1 denote the number of passages retrieved in the Stage 1 ANN, and k_4 denote the number of passages retrieved in the Stage 4 ANN. Then, for (i) the ColBERT-PRF Ranker model, we apply with rank cutoff of $k_1 = 300$ and $k_4 = 1000$[19], and for (ii) the ReRanker model, we apply with rank cutoff $k_1 = 1000$ in the first stage only, to ensure sufficient recall of relevant passages to be upranked after applying PRF. We later vary k_1 and k_4 to demonstrate their impact upon efficiency and effectiveness.

4.6.3 Results

Now we analyse the evaluation results, in response to RQ4.8 and RQ4.9, for the efficient variants of ColBERT-PRF.

4.6.3.1 RQ4.8 - Clustering Variants

Table 4.6 lists the effectiveness and the efficiency performance for ColBERT E2E and the ColBERT-PRF instantiated as Ranker and ReRanker models on both the TREC 2019 and TREC 2020 passage ranking query sets. In terms of efficiency, we measure the MRT of the different ColBERT-PRF stages as well as the overall MRT for each model variant. From Table 4.6, we note that, for both the TREC 2019 and TREC 2020 query sets, both the ColBERT-PRF Ranker and ReRanker model variants implemented with KMeans-Closest and KMedoids clustering methods are much faster than the KMeans clustering method model, without markedly compromising their effectiveness. In particular, both KMeans-Closest and KMedoids still exhibit enhanced nDCG@10 and MAP (significantly so) over the ColBERT E2E baseline. Moreover, this speed benefit is obtained by omitting the FAISS lookup step in the default ColBERT-PRF with KMeans-Closest and KMedoids clustering algorithms, as large efficiency improvements can be observed in the Stage 3 column of Table 4.6 (e.g. on TREC 2019, $\sim$900ms for KMeans-Closest vs. $\sim$3000ms for KMeans). Going further, KMedoids is faster still (218ms on TREC 2019), demonstrating the benefit of a fast clustering algorithm, with no further loss of effectiveness compared to KMeans-Closest.

Overall, in a reranking scenario, KMeans-Closest and KMedoids clustering methods experience upto 2.48× and 4.54× speedups, respectively. Indeed, the mean response times of KMedoids of 766ms (TREC 2020) is very respectable compared to the ColBERT E2E baseline, despite the normally expensive application of a PRF technique. Thus, in response to RQ4.8, we conclude that for both the ColBERT-PRF Ranker and ReRanker models with KMeans-Closest or KMedoids clustering are more efficient than the KMeans clustering method without compromising the effectiveness.

[19] Indeed, Macdonald and Tonellotto (2021) suggest $k = 300$ is sufficient for high precision retrieval.

4.6.3.2 RQ4.9 - Variants using Approximate Scoring

Next, we consider the application of Approximate Scoring within ColBERT-PRF. Again, efficiency and effectiveness results are reported in Table 4.6. We report response times only for KMeans. Firstly, on examining the table, we find that Approximate Scoring applied in both the first stage and the fourth stage of the ColBERT-PRF Ranker model exhibits similar effectiveness performance but is much more efficient than the original ColBERT-PRF Ranker model. In addition, deploying Approximate Scoring within the ColBERT-PRF ReRanker model also reduces the response time while still outperforming the ColBERT E2E model (but not by a significant margin). From Table 4.6, we see that rows with Approximate Scoring techniques applied exhibit increased Stage 1 times (43ms → 95ms/90ms for Ranker, as MaxSim takes time to compute), but are much faster in Stage 2, as the exact scoring only occurs in the selected high-quality candidates (344ms → 22ms/23ms for Ranker). The next effect of replacing both of the set retrieval ANN stages with Approximate Scoring in Ranker is an up to 18% speedup in response times (4103ms → 3466ms), while still maintaining high effectiveness, e.g., significant improvements in MAP over the baseline ColBERT E2E.

Next, we further study the trade-off between the efficiency and the effectiveness of ColBERT-PRF applied with Approximate Scoring, as well as the benefits brought by the different clustering techniques. Aligned with the table, Figure 4.12 presents both the effectiveness and efficiency of the following three strategies on the TREC 2019 query set: (i) ColBERT-PRF Ranker applied with Approximate Scoring in stage 1 using three different clustering techniques; (ii) ColBERT-PRF Ranker applied with Approximate Scoring in both stage 1 and stage 4 using three different clustering techniques and (iii) ColBERT-PRF ReRanker applied with Approximate Scoring in stage 1 using three different clustering techniques. In each figure, we vary the cutoff, k_1 or k_4, of Approximate Scoring to produce curves for each setting ($100 \leq \{k_1, k_3\} \leq 7300$[20]). We provide separate figures for MAP and nDCG@10. Each figure has two asterisk points (★) denoting the performance of ColBERT E2E, and the ColBERT-PRF default setting (KMeans, ANN set retrieval). For the points in each curve, the marker • indicates the corresponding performance is significantly improved (and × indicates not significantly) over the ColBERT E2E baseline.

Firstly, we analyse ColBERT-PRF Ranker when only Stage 1 Approximate Scoring is applied. From Figure 4.12b, we observe that, for the smaller k_1, there is some minor degradation of nDCG@10; but the impact on MAP (Figure 4.12a) is indistinguishable. In terms of efficiency, it can easily be seen that KMedoids is the most efficient technique, followed by the KMeans-Closest technique and finally the KMeans clustering technique.

We next consider Figure 4.12c and Figure 4.12d, where we applied the Approximate Scoring technique for both the first and fourth stages for ColBERT-PRF Ranker model, with the different clustering methods. More specifically, k_1, the rank cutoff of the first stage Approximate Scoring is fixed to 300, while k_4 is varied. From Figure 4.12c, we find that all of the three clustering

[20] 7300 is the average number of passages retrieved by ColBERT E2E for $k' = 1000$

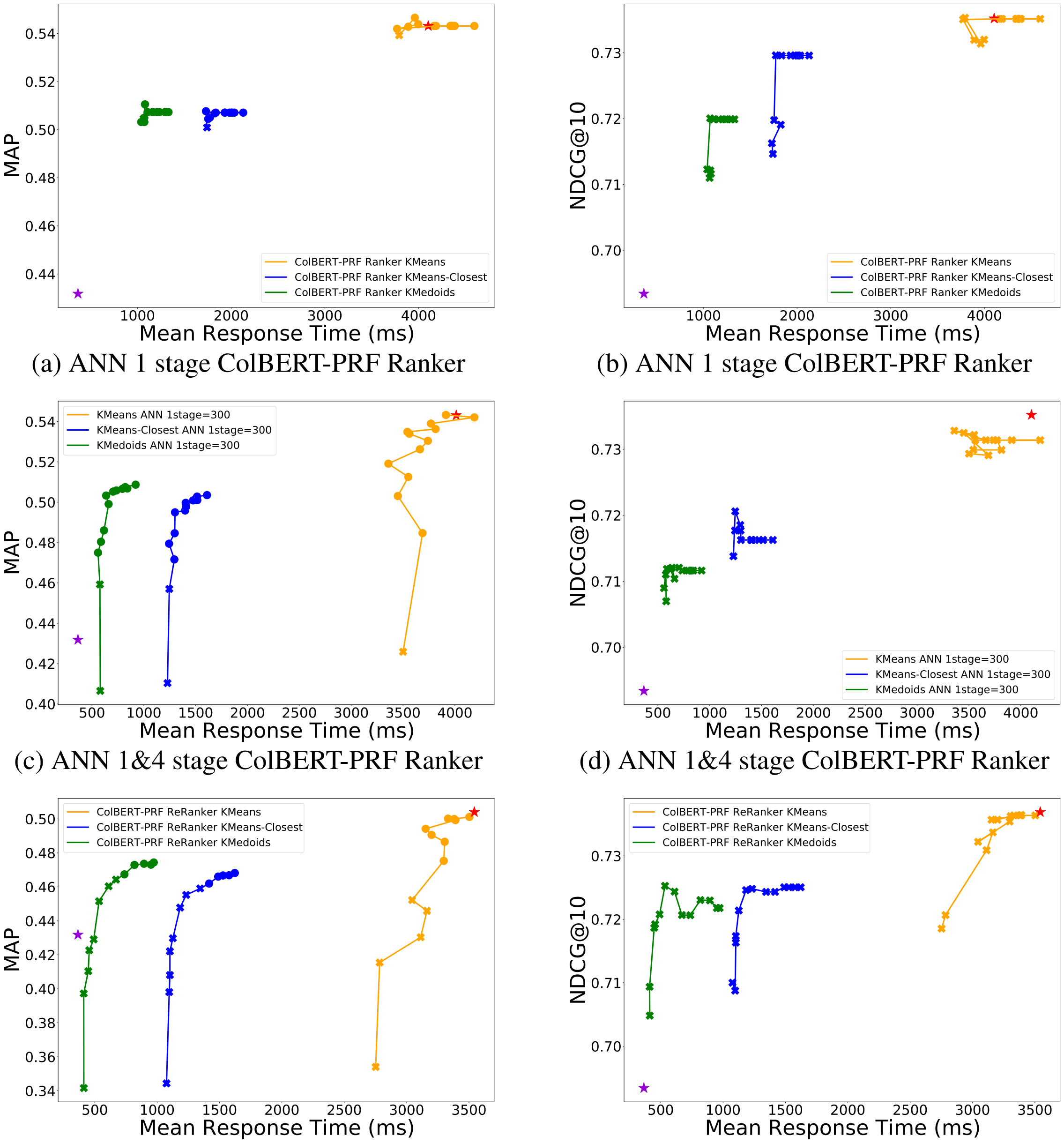
Table 4.6: Mean response time and the effectiveness on both TREC 2019 and TREC 2020 passage ranking query sets. † indicates significant improvement over the ColBERT-E2E model. The highest effectiveness and lowest response time value in each scenario is boldfaced.

Models	PRF Description	Mean Response Time (ms)						MAP	nDCG@10	MRR	Recall
		Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Overall				
					TREC 2019 query set						
ColBERT E2E	–	47	318	-	-	-	365	0.4318	0.6934	0.8529	0.7892
ColBERT-PRF Ranker	KMeans	**43**	344	2997	61	658	4103	0.5431†	**0.7352**	**0.8858**	**0.8706†**
	KMeans-Closest	45	333	903	116	641	2038 (2.01x)	0.5075†	0.7289	0.8497	0.8507†
	KMedoids	45	327	**218**	134	610	**1334 (3.07x)**	0.5073†	0.7200	0.8723	0.8681†
	Approximate Scoring (Stage 1)	95	**22**	3011	**56**	684	3868 (1.06x)	**0.5478†**	0.7314	0.8649	0.8649†
	Approximate Scoring (Stages 1 & 4)	90	23	3158	129	**66**	3466 (1.18x)	0.5196†	0.7314	0.8042	0.8646†
ColBERT-PRF ReRanker	KMeans	**47**	374	3047	-	75	3543	**0.5040†**	**0.7369**	**0.8858**	**0.7961**
	KMeans-Closest	**47**	352	921	-	110	1430 (2.48x)	0.4700†	0.7062	0.8497	0.7890
	KMedoids	**47**	351	**257**	-	139	**794 (4.46x)**	0.4744†	0.7235	0.8723	0.7892
	Approximate Scoring (Stage 1)	93	**56**	3214	-	**68**	3431 (1.03x)	0.4565	0.7336	0.8858	0.6953
					TREC 2020 query set						
ColBERT E2E	–	44	346	-	-	-	390	0.4654	0.6871	**0.8525**	0.8245
ColBERT-PRF Ranker	KMeans	**45**	346	3033	54	677	4155	**0.5116†**	**0.7152**	0.8439	**0.8837†**
	KMeans-Closest	**45**	348	945	120	647	2105 (1.97x)	0.4920†	0.7054	0.7850	0.8670†
	KMedoids	**45**	338	**222**	134	609	**1348 (3.08x)**	0.4970†	0.7065	0.8363	0.8787†
	Approximate Scoring (Stage 1)	91	**22**	3030	**60**	711	3914 (1.06x)	0.5062†	0.7108	0.8417	0.8802†
	Approximate Scoring (Stages 1 & 4)	89	**22**	3086	137	**63**	3397 (1.22x)	0.4954†	0.7091	0.8019	0.8419
ColBERT-PRF ReRanker	KMeans	47	374	2922	-	**64**	3477	**0.5049†**	**0.7165**	0.8439	**0.8246**
	KMeans-Closest	**46**	352	987	-	106	1491 (2.33x)	0.4908†	0.7061	0.7850	0.8255
	KMedoids	47	341	**251**	-	127	**766 (4.54x)**	0.4927†	0.7077	0.8363	0.8245
	Approximate Scoring (Stage 1)	96	**54**	3110	-	72	3332 (1.04x)	0.4858	0.7127	**0.8464**	0.7550

techniques exhibit correlations between efficiency and effectiveness, in that increased MRT also exhibits increased effectiveness. Moreover, reducing k_4 results in more marked degradations for MAP than for nDCG@10, and, for each of the three clustering methods, stable effectiveness can be achieved with large enough k_4.

Finally, we analyse the efficiency/effectiveness trade-off for ColBERT-PRF ReRanker. From Figures 4.12e & 4.12f, we observe that the trade-off curves for ColBERT-PRF ReRanker model with different clustering technique exhibit similar trend with Figure 4.12c & 4.12d, with the slightly lower MAP values typically exhibited by ReRanker in comparison to the Ranker setting of ColBERT-PRF. Overall, reducing k_1 here can markedly impact both MAP and nDCG@10. However, using sufficiently large k_1 can still result in significantly enhanced MAP (denoted using ●), even with response times around 1000ms. This is markedly faster than the default ColBERT-PRF ReRanker setting, which attains 3500ms (shown as ★) and much closer to the default response time of ColBERT E2E (★).

Overall, in response to RQ4.9, we conclude that the Approximate Scoring technique is useful to attain a better balance of effectiveness and efficiency for the ColBERT-PRF model, by reducing the number of documents being re-ranked, and can also be combined with the more efficient clustering techniques.



(a) ANN 1 stage ColBERT-PRF Ranker

(b) ANN 1 stage ColBERT-PRF Ranker

(c) ANN 1&4 stage ColBERT-PRF Ranker

(d) ANN 1&4 stage ColBERT-PRF Ranker

(e) ANN 1 stage ColBERT-PRF ReRanker

(f) ANN 1 stage ColBERT-PRF ReRanker

Figure 4.12: Trade-off between efficiency and effectiveness for ColBERT-PRF implemented with different clustering methods and the Approximate Scoring technique. The star coloured with purple and the red represents the ColBERT E2E and the default ColBERT PRF Ranker or ReRanker performance. A point marker of • indicates the corresponding performance is significantly improved (and × indicates not significantly) over the ColBERT-E2E baseline.

4.7 Conclusion

In this chapter, we proposed a contextualised pseudo-relevance feedback mechanism for multiple representation dense retrieval. Based on the feedback documents obtained from the first-pass retrieval, our proposed ColBERT-PRF approach extracts representative feedback embeddings using a clustering technique. It then identifies discriminative embeddings among these representative embeddings and appends them to the query representation. ColBERT-PRF can be effectively applied in both ranking and reranking scenarios, and requires no further neural network training beyond that of ColBERT. Indeed, our passage ranking experimental results – on the TREC 2019 and 2020 Deep Learning track passage ranking query sets (cf. Table 4.1) – showed that our proposed approach can significantly improve the retrieval effectiveness of the state-of-the-art ColBERT dense retrieval approach. In particular, our ColBERT-PRF outperforms ColBERT E2E model by 26% and 10% on TREC 2019 and TREC 2020 passage ranking query sets (cf. Table 4.4). Our proposed ColBERT-PRF is a novel and extremely promising approach into applying PRF in dense retrieval. It may also be adaptable to further multiple representation dense retrieval approaches beyond ColBERT. We further validated the effectiveness of the proposed ColBERT-PRF approach on the MSMARCO document ranking task and TREC Robust04 document ranking task, where ColBERT-PRF is observed to exhibit upto 21% and 14% improvements over ColBERT E2E model on TREC 2019 and TREC 2020 document ranking query sets, respectively. Moreover, we investigated ColBERT-PRF variants with different weighting approaches for measuring the usefulness of the expansion embeddings. Finally, in order to trade-off the efficiency and the effectiveness, we explored the efficient variants of ColBERT-PRF using the approximate scoring technique and/or different clustering algorithms, bringing upto 4.54x speedup without compromising the retrieval effectiveness (cf. Table 4.6). In conclusion, the main findings of this chapter can be summarised as follows:

- The pseudo-relevance feedback information from the top-returned documents in multiple representation dense retrieval is beneficial for improving the retrieval effectiveness on passage retrieval (cf. Section 4.3) and document retrieval (cf. Section 4.4). Indeed, our proposed pseudo-relevance feedback mechanism can significantly improve the retrieval effectiveness over than ColBERT end-to-end model, the single representation dense retrieval models, as well as most of the baselines for both passage ranking and document ranking tasks;

- Techniques based on statistical information, namely IDF and ICTF, and on embedding coherency, namely Mean Cosine Similarity, can be used to measure the informativeness of expansion embeddings of ColBERT-PRF (Section 4.5);

- The trade-off of the retrieval effectiveness and efficiency of ColBERT-PRF can be attained using different clustering techniques and/or candidate selection techniques based on approximate scoring (Section 4.6).

Overall, in the proposed thesis statement in Section 1.1, we postulated that applying pseudo-relevance feedback on contextualised token embeddings can refine the query representation for multiple representation dense retrieval. Therefore, in this chapter, we explored how to implement the pseudo-relevance feedback mechanism on ColBERT and proposed a method named ColBERT-PRF to produce a refined query representation for effective dense retrieval.

Our proposed ColBERT-PRF makes it feasible to implement the pseudo-relevance feedback technique in a multiple-representation dense retrieval setting. In particular, the provided extensive experimental results demonstrate the effectiveness of our proposed ColBERT-PRF model.

However, one of the limitations of this chapter is that it only examines the in-domain retrieval effectiveness of ColBERT-PRF models. Indeed, most of the dense retrieval models, such as ANCE and ColBERT models, are shown to face challenges when it comes to out-of-domain evaluations due to the knowledge shift between different domains. Therefore, in Chapter 5, we investigate the out-of-domain effectiveness of dense-PRF models. Specifically, we use high-quality external knowledge to assist in achieving effective dense external expansion. Moreover, we note that the default ColBERT model applies only to BERT and WordPiece tokeniser. However, the effect of the pretrained model and the tokenisation method for the contextualised late interaction mechanism used by ColERT is still under investigation. Therefore, in Chapter 6, we extend ColBERT to Col$\star$ and ColBERT-PRF to Col$\star$-PRF, by generalising the de-facto standard BERT PLM to various different PLMs. Moreover, we also investigate the impact of different tokenisation techniques on the nature of matches occurring among Col$\star$ and Col$\star$-PRF models in Chapter 6. To this end, the ColBERT-PRF as well as the Col$\star$-PRF models perform dense query expansion in an unsupervised manner and might be affected by heuristic techniques such as clustering and IDF statistics. Thus, in Chapter 7, we explore a new model that performs the dense query expansion in a supervised way. In particular, we propose a contrastive weighting model to learn to assign high weights for PRF tokens that can better discriminate the relevant documents from the non-relevant documents.

Chapter 5

Dense External Expansion

In Chapter 4, we validated the second hypothesis in our proposed thesis statement (cf. Section 1.1) by proposing a method called ColBERT-PRF, which implements the pseudo-relevance feedback mechanism on the multiple representation dense retrieval paradigm. In particular, similar to the various pseudo-relevance feedback techniques in the literature (cf. Section 2.4), ColBERT-PRF refines the input query representation based on the pseudo-relevance feedback information. However, if the quality of the pseudo-relevance feedback documents is limited, for instance, the limited vocabulary for the traditional sparse-PRF or the limited document representations for the dense-PRF techniques may not help identify more relevant documents. Indeed, especially for some hard queries, the feedback set may contain non-relevant documents, thus causing topic drift from incorrect expansion terms. Therefore, we postulate the third hypothesis posed in our thesis statement (cf. Section 1.1) and ask that: *can performing external pseudo-relevance feedback based query reformulation improve the zero-shot retrieval for both sparse and dense retrieval?* Indeed, it has previously been shown that high-quality external corpora can also produce feedback documents with complementary information (Diaz and Metzler, 2006, Kwok and Chan, 1998, Peng et al., 2009a,b, Xu et al., 2009)[1]. Thus, different query-related words can be extracted to reformulate, typically expand, the original query. Usually, the additional collection is referred to as the *external collection* while the local collection used is referred to as the *target collection*. However, such *external expansion* approaches have only been studied for sparse retrieval methods (cf. Section 2.1), and their effectiveness for recent dense retrieval methods (cf. Section 2.3) remains uninvestigated.

Indeed, as we mentioned in Section 2.3, as the training of such dense retrieval models requires large amounts of training data, there may not be enough training data for smaller corpora to train effective domain-specific dense retrieval models. An attractive solution is to *transfer* sufficiently trained dense retrieval models from large corpora to smaller domains. For instance, MacAvaney et al. (2019) showed that an effective BERT reranker could be trained using the MSMARCO

[1] This technique is also known as *collection enrichment* (Kwok and Chan, 1998); we prefer the modern nomenclature of *external expansion* (Diaz and Metzler, 2006), which is more easily relatable to its definition.

passage ranking data, for adhoc search, as well as COVID-related literature (MacAvaney et al., 2020a). In contrast, attempts using zero-shot dense retrieval models have underperformed compared to existing models (Chen et al., 2022, Thakur et al., 2021), which can also be supported by the findings of the experiments discussed in the previous chapter, Section 4.4. Therefore, to mitigate this domain shift between the external trained collection and the target collection which has been discussed in Section 2.5.3, in this chapter, we propose to employ the external expansion from the external high-quality collection thus improving zero-shot retrieval performance.

On the one hand, we investigate the benefit of the classical sparse PRF models, as introduced in Section 2.4.1, that is supported by a pseudo-relevant feedback set obtained from different dense retrieval approaches. Moreover, we study the effectiveness of external dense expansion on dense retrieval. This facilitates an investigation into how external expansion changes the semantic manner of retrieval. Finally, we investigate whether the sparse external retrieval can produce feedback information that is useful for dense retrieval.

In summary, this chapter makes the following contributions:

- We investigate external expansion when mixing sparse & dense retrieval paradigms (including both single representation and multiple representation dense retrieval); sparse and zero-shot dense retrieval experiments are conducted on two classical TREC test collections (Robust04 & WT10G);
- We also conduct the zero-shot evaluation on four BEIR datasets (Thakur et al., 2021) (namely DBPedia, NFCorpus, TREC-COVID and Touché-2020) - a set of datasets selected for evaluating zero-shot evaluation;
- We deploy two sparse weighting models (BM25 & DPH), two sparse PRF approaches (RM3 & Bo1), two dense retrieval models (ANCE & ColBERT), and two dense RPF approaches (ANCE-PRF & ColBERT-PRF);
- We analyse the propensity for a multiple representation PRF technique to perform semantic vs. exact token matching, under normal and external PRF conditions.

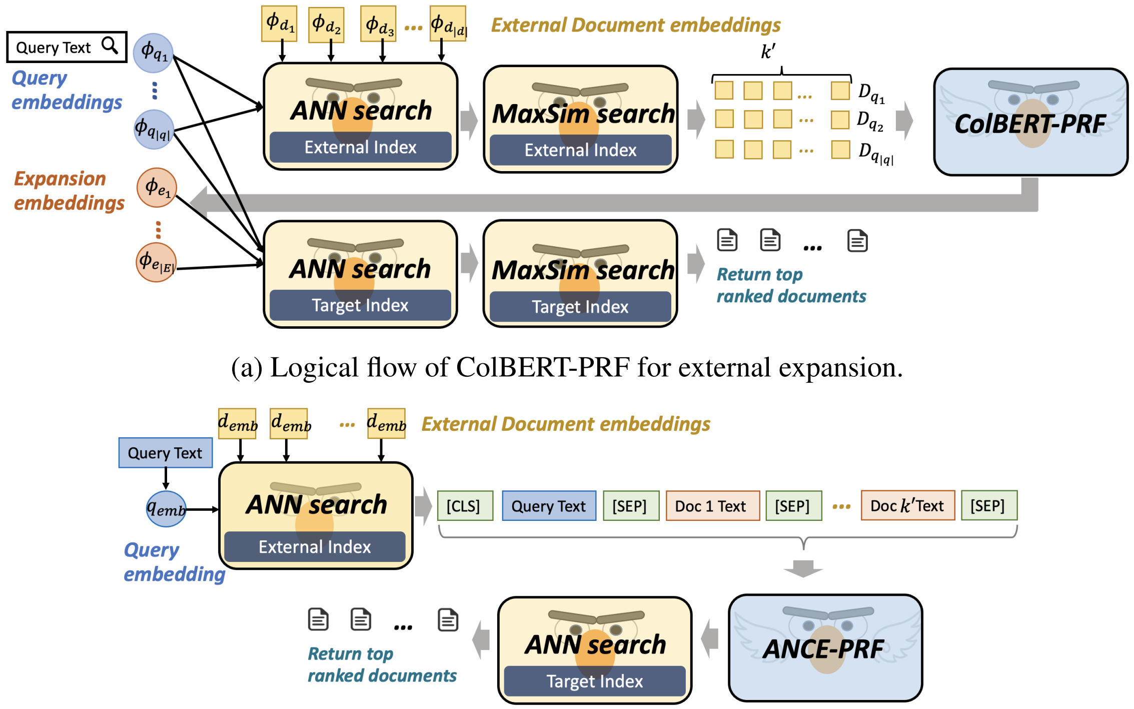
The main findings of this chapter can be summarised as follows: (i) high-quality feedback documents obtained using multiple representation dense retrieval, namely ColBERT, from a high-quality external collection can significantly improve sparse retrieval for both Robust04 (by 12% for nDCG@10) and WT10G (by 28% for nDCG@10); (ii) significant sparse retrieval effectiveness improvements are also observed when performing expansion using external feedback documents obtained using a single representation dense retrieval, namely ANCE; (iii) extracting PRF documents from an external collection using ColBERT or ANCE for dense retrieval can significantly outperform the zero-shot dense retrieval models on target, indicating the utility of the dense external expansion to improve the effectiveness of zero-shot dense retrieval.

The remainder of this chapter is structured as follows: Section 5.1 discusses related works of the external expansion technique; Section 5.2 instantiates this framework for external pseudo-relevance expansions Section 5.3 elicits our research questions; Section 5.4 and Section 5.5 present the experimental setup and results of this work. Finally, we provide the concluding remarks in Section 5.6.

5.1 Related Work on External Expansion

Instead of only obtaining the pseudo-relevance feedback documents from the target corpora, another thread of related research focuses on bringing in more high-quality feedback documents from an external corpora. For instance, Kwok and Chan (1998) proposed an external expansion approach, which borrowed similar documents from an external collection when the quality of the initial returned documents is poor, in order to improve PRF. External expansion techniques were popular in the top runs in the TREC Robust Track (Voorhees, 2005, 2006). Indeed, Diaz and Metzler (2006) studied the collection size and other characteristics that external corpora should have to provide useful external information for a target collection. Meanwhile, Peng et al. (2009a,b) further studied a selective external expansion approach based on the query performance prediction, where external expansion was only applied when the external corpus was predicted to better answer the query than the target corpus. However, these external corpora based query reformulation techniques are only deployed in the classical sparse retrieval paradigm and have not yet been explored for the newer dense retrieval paradigm.

More recently, initial investigations have taken place into how pseudo-relevance feedback information can improve the effectiveness of dense retrieval. For instance, ColBERT-PRF identifies representative and important embeddings from the pseudo-relevant set to expand the original query embeddings. Based on the experimental results in Chapter 4, ColBERT-PRF (cf. Section 4.2) was shown to be very effective, exhibiting performances as high as the top-ranked group in terms of MAP and nDCG@10 in the TREC 2019 Deep Learning track. In contrast, instead of performing query expansion, ANCE-PRF (cf. Section 2.4.3), employs the pseudo-relevance feedback information within a retrained query encoder to capture the feedback information while encoding the query into a single high-dimensional embedding. Both ColBERT-PRF and ANCE-PRF aim to capture the useful information from the pseudo-relevance feedback documents to obtain query representations of the two PRF models, however, they differ in the exact query representation, by using multiple representation and single representation query embeddings, respectively. Therefore, in this chapter, we build on top of the ColBERT-PRF and ANCE-PRF models, respectively, to investigate the application of external expansion in PRF for dense retrieval. In addition, we also study external expansion in a hybrid framework of the dense (ANCE and ColBERT) and sparse retrieval approaches.



(a) Logical flow of ColBERT-PRF for external expansion.

(b) Logical flow of ANCE-PRF for external expansion

Figure 5.1: Dense PRF variants for external expansion.

Table 5.1: Summary of the different retrieval and PRF processes used in this work.

Process	Inputs	Outputs
	Retrievers ($Q \rightarrow R$)	
BM25	Query Text	Retrieved Documents
DPH	Query Text	Retrieved Documents
ColBERT	Query Text or Multi-Rep. Query Embs.	Retrieved Documents
ANCE	Query Text or Single-Rep. Query Emb.	Retrieved Documents
	PRF ($R \rightarrow Q$)	
RM3	Query Text & Retrieved Documents Text	Query Text w/ weights
Bo1	Query Text & Retrieved Documents Text	Query Text w/ weights
ColBERT-PRF	Orig. Query Embs. & Retrieved Document Embs.	Multi-Rep. Query Embeddings w/ weights
ANCE-PRF	Query Text & Retrieved Documents	Single-Rep. Query Embedding

5.2 Our Proposed Method: Dense External Expansion

External expansion (Diaz and Metzler, 2006), also known as collection enrichment (Kwok and Chan, 1998), is a classical PRF-based technique for improving retrieval effectiveness given corpus. In particular, PRF is known to fail if the quality of the feedback documents are poor (Kwok and Chan, 1998, Peng et al., 2009a,b, Xu et al., 2009). To address this, in external expansion, a separate high-quality *external* index is used for an initial retrieval. By obtaining a pseudo-relevant feedback set from the external index, it is assumed to identify additional and higher quality expansion terms than might be found in a pseudo-relevant feedback set obtained on the *target* corpus. This is particularly useful if the top-ranked documents on the target corpus are homogeneous in their choice of vocabulary, or if there are few documents containing the original query terms. By expanding the query using an external corpus, the higher diversity of expansion terms can result in more relevant documents being retrieved in the target corpus. In summary, we hypothesise that external expansion can help to address the domain shift when transferring a dense retrieval model towards a target collection in a zero-shot fashion.

Using the notation introduced in Section 2.1.2, where we used $Ret(I,k)(q)$ to represent a retrieval process performing on the index I and return k documents in response to the input query q. Based on this, external expansion using a target index I_{target} and an index of a higher quality corpus, I_{ext}, can be expressed as follows:

$$\texttt{Ret}_{\texttt{BM25}}(I_{ext},k') \overset{R}{\gg} \texttt{PRF}_{\texttt{RM3}}(I_{ext},\theta) \overset{Q}{\gg} \texttt{Ret}_{\texttt{BM25}}(I_{target},k). \tag{5.1}$$

Based on this, we propose that the dense PRF models, namely ColBERT-PRF and ANCE-PRF, can be instantiated for external expansion, operating entirely in a dense retrieval space:

$$\texttt{Ret}_{\texttt{Dense}}(I_{ext},k') \overset{R}{\gg} \texttt{PRF}_{\texttt{Dense}}(I_{ext},\theta) \overset{Q}{\gg} \texttt{Ret}_{\texttt{Dense}}(I_{target},k), \tag{5.2}$$

where the subscript `Dense` can be instantiated as `ColBERT` or `ANCE`, considering whether the dense retrieval is performed in multiple representation or single representation embedding spaces. If instantiated as `ColBERT`, a set of expanded query embeddings are obtained from an index of a high-quality external corpus, but are then executed on the index of the target corpus, I_{target}. Figure 5.1a shows the logical flow of query embeddings for ColBERT-PRF in an external expansion setting. In contrast, when instantiating as `ANCE`, a refined query representation is encoded using the high-quality feedback documents from the external corpus. The logical flow of ANCE-PRF for external expansion is depicted in Figure 5.1b.

To the best of our knowledge, this is the first application of external expansion in a dense retrieval space. For both ColBERT-PRF and ANCE-PRF models, external expansion is possible as long as the underlying encoders used for both external and target retrieval – and expansion in the case of ANCE-PRF – are unchanged. This ensures that the embedded query representations generated from the external corpus can be used to retrieve documents on the target corpus.

A natural question that arises is as follows: *If PRF$_{Dense}$ already takes contextualised information from the feedback documents into account (thereby addressing word mismatch and polysemy), why is external PRF$_{Dense}$ necessary?* We answer this by arguing that a high-quality external corpus, such as Wikipedia, can contain more broader information about the topic, which results in more diverse and valuable expansion embeddings than the local corpus. Moreover, as the expansion embeddings are contextualised, there is less risk of topic drift, as might occur for a classical term-based feedback approach. Similarly, for the ANCE-PRF side, the high-quality external corpus contains feedback documents that may have a more diverse description of the query topic, thus further enriching the embedded query representation. On the other hand, using the notation in Equation (5.2), it is clear to see that other "hybrid" formulations are possible.

Indeed, while the PRF mechanism is tightly coupled (e.g. RM3 generates term-based queries, suitable for the sparse retrieval models such as BM25; ColBERT-PRF generates embedding queries, suitable for ColBERT)[2], the first-stage retrieval can be varied independently. This allows us to vary the choice of first stage ranker – for instance, sparse or multi-representation dense (i.e. ColBERT) vs. single-representation dense (i.e. ANCE).

Indeed, in considering conventional and external PRF configurations in this manner, we are able to determine (a) the impact of dense retrieval on identifying useful feedback documents that may have a minimal lexical match with the original query, (b) the value of an external corpus, (c) the benefit of a dense-based PRF technique.

Hence, based on the notations introduced in Section 2.1.2 and the established various retrieval models, from the sparse retrieval models in Section 2.1 to the dense retrieval models in Section 2.3, as well as the PRF retrieval pipelines in Section 2.4, Table 5.1 lists the inputs and the corresponding outputs for the retrievers (top half of the table) and the PRF models (bottom half). For instance, among the retrieval models, the inputs to the sparse retrieval models, namely BM25 and DPH, are different from those of the dense retrieval models, ColBERT and ANCE. In particular, the input query text is encoded into multiple query embeddings for ColBERT and into a single query embedding for ANCE. In addition, for the PRF techniques, we see that there are variations in the inputs and outputs are different among the sparse and dense PRF models. Both RM3 and Bo1 take the query text and pseudo-relevance feedback document text as input and produce an expanded query, i.e. words with weights. On the other hand, ColBERT-PRF takes original query embeddings and the embeddings of the feedback document as input and outputs a list of expansion embeddings with weights. In contrast, the ANCE-PRF's inputs are the text of the query and of the feedback documents and its output is a refined single representation query embedding. In this way, it is clear that the PRF and subsequent retrieval stages are tightly coupled – they cannot be mismatched; however it is possible to change the retrieval stage preceding PRF - either to a different retrieval model, or even to a different index.

5.3 Research Questions

This chapter focuses on improving the performance of zero-shot dense retrieval models. In particular, we investigate the external pseudo-relevance feedback based on dense retrieval feedback information, where the similarity search is conducted based on contextualised information rather than the statistical information used by traditional sparse retrieval models.

Our experiments first study the effectiveness of applying and extracting terms (i.e. sparse feedback) based on external feedback passages that have been identified by a contextualised retrieval model. Thus, we pose our three research questions as follows:

[2] Note that the reverse configurations are not tested – for instance, RM3 returns a bag of weighted terms, which would not have sufficient contextual information to be accurately encoded by ColBERT.

Table 5.2: Summary of the main configurations for each of the research questions.

	Baselines		Treatments	
Research Questions	Stage 1 (Target)	Stage 2 (Target)	Stage 1 (External)	Stage 2 (Target)
RQ5.1	Sparse	Sparse	Dense or Sparse	Sparse
RQ5.2	Dense	Dense	Dense	Dense
RQ5.3	Sparse	Dense	Sparse	Dense

RQ5.1: What is the benefit of using dense retrieval in obtaining external feedback documents for sparse retrieval?

In addition, we investigate the performance of the zero-shot dense retrieval models using dense external expansion, in both single representation and multiple dense representation-based paradigms. Accordingly, we propose the following second research question.

RQ5.2: What is the benefit of using dense retrieval, in the form of (a) multiple representation (ColBERT) & (b) single representation (ANCE), in obtaining external feedback documents for dense retrieval?

Next, we investigate the impact of the initial stage retrieval for the dense pseudo-relevance feedback models, as follows:

RQ5.3: Do the dense pseudo-relevance feedback models require dense retrieval to obtain their feedback documents?

Table 5.2 summarises the configurations of 1st and 2nd stage retrieval paradigms for the baselines and treatments that are tested for each research question.

5.4 Experimental Setup

In the following, we describe the external and target datasets used in this work in Section 5.4.1. Then the detailed implementation setup and baselines are discussed in Section 5.4.2 and Section 5.4.3. Finally, the metrics used in our experiments are detailed in Section 5.4.4.

5.4.1 Datasets

Table 2.4 in Chapter 2 describes the detailed collection statics for both target collections and external collection used in this work.

Target Collections: In this work, we evaluate using two TREC adhoc test collections, namely the Robust04 and the WT10G as detailed in Section 2.5.1. For evaluation, we use the title-only and also the (longer) description queries, 250 from Robust04 and 100 from WT10G. In addition, we evaluate on four BEIR datasets, which are detailed in Section 2.5.1, namely DBPedia, NFCorpus, TREC-COVID and Touché-2020 .

External Collection: The external collection used in this work is the MSMARCO passage corpus (cf. Section 2.5.1), which contains approximately 8.8M passages. Indeed, past work (Peng et al., 2009a, Xu et al., 2009) have successfully used Wikipedia as an external corpus, due to its encyclopedic nature. We use MSMARCO as introduced in Section 4.3.2, as it contains high-quality passages about a number of topics, including from Wikipedia (Nguyen et al., 2016)[3].

5.4.2 Implementation and Settings

All the experiments in this work are conducted on the PyTerrier IR experimentation platform (cf. Section 2.1.2). We build the ColBERT dense indices following the same settings used in Section 4.3.2.2, such as padding queries upto a length of 32 tokens, and truncating passages to 180 tokens. We employ the ColBERT model checkpoint and the implementation of the ColBERT-PRF model used in Chapter 4 (which was trained on the MSMARCO passage training dataset), and follow the ColBERT-PRF default parameter settings reported in Section 4.3 including using 3 feedback passages and 10 expansion embeddings.[4] In addition, we build the ANCE dense indices using the checkpoint released by the authors (Xiong et al., 2021). Similar to ColBERT, ANCE, as well as the ANCE-PRF model, have also been trained using the MSMARCO passage training dataset. For ANCE-PRF, we experiment with the author (Yu et al., 2021b) provided model checkpoint trained with 3 feedback passages.[5] Thus, we also use 3 feedback passages for the ANCE-PRF experiments. For all external expansion experiments, we mix the source from external and target corpus. For Robust04 and WT10G, we apply passaging (applying a sliding window of length 150 tokens and stride 75), and documents are ranked by applying max passage. ANCE dense indices are created using model checkpoints released by the original ANCE authors (Xiong et al., 2021), which were also trained on the MSMARCO passage task. For sparse PRF models, we employ both RM3 and Bo1 techniques (cf. Section 2.1) and follow the default parameter settings of PyTerrier – i.e. 3 feedback passages and 10 expansion terms. For sparse retrieval, we use the stemmed sparse index built using PyTerrier. Our virtual appendix contains the result files of all experiments, and the notebooks needed to reproduce these experiments.[6]

5.4.3 Baselines

To test the effectiveness of our external pseudo-relevance expansion technique approach, we compare it with the following baselines:

- **Sparse Approaches:** We apply sparse retrieval models without PRF, namely the BM25 and DPH weighting models. We also combine these models with sparse pseudo-relevance expansion technique on the target inverted index, i.e. BM25 with RM3 PRF (cf. Section 2.4.1)

[3] Our initial experiments found that performing external expansion using a number of other corpora did not improve retrieval effectiveness on MSMARCO.

[4] `github.com/terrierteam/pyterrier_colbert`

[5] `github.com/yuhongqian/ANCE-PRF`

[6] `github.com/Xiao0728/DenseExternalExpansion_VirtualAppendix`

& DPH with Bo1 PRF (cf. Section 2.4.1). Furthermore, we also instantiate these PRF models in an external expansion setting, i.e. $Ret_{BM25}(ext) \overset{R}{\gg} PRF_{RM3}(ext) \overset{Q}{\gg} Ret_{BM25}(target)$ & $Ret_{DPH(}ext) \overset{R}{\gg} PRF_{Bo1}(ext) \overset{Q}{\gg} Ret_{BM25}(target)$.

- **Neural Reranking Approaches:** To aid in our comparisons, we further apply neural rerankers, namely ColBERT and ANCE reranking models upon the sparse retrieval models. For instance, applying a final ColBERT reranker upon BM25 with RM3 query expansion would be denoted as: $Ret_{BM25} \overset{R}{\gg} PRF_{RM3} \overset{Q}{\gg} Ret_{BM25} \overset{R}{\gg} ColBERT$.

- **Dense Approaches:** We also deploy dense retrieval models, with and without a pseudo-relevance feedback mechanism. In particular, for single representation dense retrieval, we deploy ANCE (cf. Section 2.3.1)), and for multiple representation dense retrieval (cf. Section 2.3.2)), we use the ColBERT-E2E model. For dense retrieval, we apply ColBERT-PRF, which is proposed in Chapter 4, on both normal and external expansion settings.

5.4.4 Evaluation Metrics

We measure effectiveness for the scenarios described in Section 5.4.1 using the evaluation metrics introduced in Section 2.5, in particular, in terms of Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR), as well as the normalised discounted cumulative gain calculated to rank depth 10 and 20. Moreover, for each model, we compare the Recall calculated to rank depth 1000. When comparing with baseline models, we use the paired t-test ($p < 0.05$) and apply the Holm-Bonferroni multiple testing correction, as per best practices in information retrieval (Sakai, 2021).

5.5 Results

We address the posed research questions in Sections 5.5.1 - 5.5.3. We summarise the overall findings in Section 5.5.4, and in Section 5.5.5 provide an analysis of how ColBERT-PRF performs matching.

5.5.1 RQ5.1: Dense External Expansion for Sparse Retrieval

Firstly, we examine the effectiveness of the obtained external feedback for sparse retrieval. In Table 5.3, for four query sets (Robust04 title-only, Robust04 description-only, WT10G title-only and WT10G description-only), we report results when external retrieval is performed using sparse models (BM25 & DPH), as well using dense retrieval (ANCE & ColBERT). In the table, columns are used to show the processes for each retrieval stage. On each test collection, we first report the performance of the four sparse retrieval baselines, including the BM25, BM25 with RM3 query expansion model, DPH, and DPH with Bo1 query expansion model. Then, we measure the retrieval

Table 5.3: External expansion for sparse retrieval. The top half table presents the results for Robust04 query sets and the bottom half table presents the results for WT10G query sets. Superscripts a-f denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model(s). The highest value for a query set is boldfaced.

	1st $\overset{R}{\gg}$	PRF $\overset{Q}{\gg}$	2nd	Robust04 (T)					Robust04 (D)				
				MAP	nDCG@10	nDCG@20	MRR	Recall	MAP	nDCG@10	nDCG@20	MRR	Recall
				Target Retrieval Only									
(a)	BM25	-	-	.241	.432	.406	.654	.687	.245	.437	.411	.680	.690
(b)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$	.275	.447	.429	.641	.738	.276	.444	.422	.625	.738
(c)	DPH	-	-	.250	.449	.421	.670	.698	.231	.430	.400	.696	.669
(d)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$	**.284**	.462	.443	.654	**.752**	.277	.468	.440	.689	.756
				External Expansion: Sparse External Sparse Retrieval									
(e)	BM25$_{ext}$	RM3$_{ext}$	BM25$_{target}$	.270^{ac}	.446^{a}	.429^{a}	.632	.731^{ac}	.282^{acd}	.464^{abc}	.436^{abc}	.651^{b}	.748^{ac}
(f)	DPH$_{ext}$	Bo1$_{ext}$	DPH$_{target}$	.278^{ac}	.460^{a}	.438^{ac}	.661^{b}	.740^{ac}	.281^{ac}	.470^{abc}	.439^{abc}	.680^{b}	.750^{ac}
				External Expansion: Dense External Sparse Retrieval (Ours)									
	ANCE$_{ext}$	RM3$_{ext}$	BM25$_{target}$	.267^{ab}	.468^{ae}	.440^{ac}	.697^{bef}	.728^{ac}	**.295**^{ac}	. **501**^{abce}	**.470**^{abce}	**.752**^{bef}	**.767**^{ac}
	ANCE$_{ext}$	Bo1$_{ext}$	DPH$_{target}$	.274^{ac}	.467^{a}	.443^{ac}	.715^{abcdef}	.736^{ac}	.280^{ac}	.490^{abcef}	.456^{abcef}	.743^{bef}	.737^{ac}
	ColBERT$_{ext}$	RM3$_{ext}$	BM25$_{target}$	.270^{ac}	.471^{ae}	.441^{a}	.700^{bef}	.729^{ac}	.287^{acd}	.489^{abe}	.458^{abce}	.722^{be}	.759^{ac}
	ColBERT$_{ext}$	Bo1$_{ext}$	DPH$_{target}$	.277^{ac}	**.482**^{acef}	**.455**^{acdef}	**.723**^{abcdef}	.734^{ac}	.280^{acd}	.485^{abce}	.453^{abc}	.738^{abef}	.742^{ac}
				WT10G (T)					WT10G (D)				
				Target Retrieval Only									
(a)	BM25	-	-	.189	.324	.323	.555	.693	.182	.343	.333	.614	.677
(b)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$	.202	.328	.326	.505	.711	.218	.379	.358	.584	.751
(c)	DPH	-	-	.206	.342	.333	.576	.698	.187	.358	.334	.604	.670
(d)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$	.235	.364	.364	.574	.752	.230	.369	.358	.622	.748
				External Expansion: Sparse External Sparse Retrieval									
(e)	BM25$_{ext}$	RM3$_{ext}$	BM25$_{target}$	.209^{ac}	.354^{b}	.341	.543^{b}	.729^{abc}	.233^{abc}	.391^{ab}	.383	.620^{abc}	.761^{ac}
(f)	DPH$_{ext}$	Bo1$_{ext}$	DPH$_{target}$	.236^{cbcd}	.383^{abc}	.368^{abc}	.606^{b}	.740^{abc}	.250^{abc}	.405^{abc}	.398^{abcd}	.666^{b}	**.786**^{acd}
				External Expansion: Dense External Sparse Retrieval (Ours)									
	ANCE$_{ext}$	RM3$_{ext}$	BM25$_{target}$	.238^{abce}	**.420**^{abcef}	.395^{abce}	.711^{abcdef}	.753^{a}	.251^{abc}	**.534**^{abcde}	.420^{abcde}	.707^{abcde}	.779^{ac}
	ANCE$_{ext}$	Bo1$_{ext}$	DPH$_{target}$	.249^{abce}	.418^{abcef}	.400^{abcdef}	**.721**^{abcdef}	**.782**^{abc}	.252^{abc}	.425^{abcde}	.409^{abcd}	.683^{bce}	.783^{acd}
	ColBERT$_{ext}$	RM3$_{ext}$	BM25$_{target}$	.242^{abce}	.418^{abcef}	.400^{abcef}	.695^{abcdef}	.727^{a}	**.257**^{abce}	.444^{abcde}	**.428**^{abcde}	**.721**^{abcde}	.773^{ac}
	ColBERT$_{ext}$	Bo1$_{ext}$	DPH$_{target}$	**.255**^{abcef}	.417^{abce}	**.402**^{abcdef}	.687^{abcdef}	.768^{abc}	.256^{abc}	.436^{abcde}	.415^{abcde}	.686^{bce}	.776^{acd}

effectiveness of the traditional external expansion models on the sparse retrieval and the external expansion models for sparse retrieval but with feedback documents obtained using dense retrieval. On analysing Table 5.3, firstly, we compare the performance of the external expansion models on sparse retrieval using RM3 and Bo1 query expansion models with the sparse retrieval models without any PRF mechanism applied and the sparse PRF models applied only on the target collection on both Robust04 and WT10G test query sets. We notice that both RM3 and Bo1-based external expansion models on sparse retrieval can significantly improve over the models without the external expansion technique applied, which attests to the usefulness of the MSMARCO passage as an external collection for both the Robust04 and WT10G corpora. Secondly, we examine the performance of the external expansion with passages produced by dense retrieval. There are four models reported under this external expansion scheme. We observe that the highest values for most of the metrics reported are given by the external expansion models. Similarly, all the external expansion models with dense retrieved passages are significantly improved over all the baselines without the external expansion technique applied. When comparing the external expansion models combining dense retrieval with sparse PRF, we find that the former models can significantly improve over the traditional external expansion models using sparse retrieval, which indicates the superiority of the feedback documents produced by dense retrieval viz. the feedback documents produced by the sparse retrieval, e.g. compared to baselines (e) & (f) in Table 5.3. Across the query sets, we note that among the external expansion models using dense retrieval, for the title query type, ColBERT is more effective than ANCE, while for the description query type, no obvious pattern emerges among the single and multiple dense retrieval models. This suggests that ColBERT is more suitable for the title (keyword) queries, perhaps due to its token-level embeddings, rather than the single embedding of ANCE.

Overall, in answer to RQ5.1, we observe that external expansion models supplied with feedback documents obtained from dense retrieval models can bring more benefits for title-only queries.

5.5.2 RQ5.2: Dense External Expansion for Dense Retrieval

Next, we analyse the effectiveness of external expansion using both ColBERT-PRF and ANCE-PRF, in Sections 5.5.2.1 and 5.5.2.2, respectively.

5.5.2.1 RQ5.2(a): Dense Expansion on Multiple Representation Dense Retrieval

We now analyse the performance of the external expansion for dense retrieval models, where the pseudo-relevance feedback information is obtained using dense retrieval models followed by the ColBERT-PRF contextualised expansion technique. Table 5.4 reports the results of the external expansion dense retrieval models as well as the sparse query expansion models, the dense retrieval model without any query reformulation techniques applied and the dense retrieval models with ColBERT-PRF applied.

From Table 5.4, we observe that the dense external expansion models give the highest value for all the metrics on both Robust04 and WT10G title and description query sets. Indeed, ColBERT-PRF improves over ColBERT end-to-end, verifying the results of Section 4.3 & 4.4 on these smaller document corpora. We also find that ColBERT end-to-end dense retrieval model exhibits lower performance than the two sparse query expansion models on both the Robust04 and WT10G query sets – this may indicate underfitting for the title-only (keyword) and description query formulations of Robust04 and WT10G, which differs from the "question-style" of the MSMARCO passage dataset used to train the ColBERT model.

Next, we note that external expansion can significantly improve over all the dense baselines across all the metrics and significantly improve over sparse query expansion models, i.e. (a) and (b) baselines in Table 5.4, in terms of nDCG for title-only queries and MAP for the description queries. One interesting observation is that, although Recall obtained by applying external expansion using ColBERT-PRF outperforms that of both ColBERT end-to-end and ColBERT-PRF performed on the target corpus, it is still lower than the sparse expansion methods. Indeed, this explains the popular practice of applying a more expensive reranker on top of the sparse retrieval models rather than on top of the dense expansion models. Moreover, when we look back to compare the external expansion dense model with the external expansion sparse model but using dense retrieved passage models in Table 5.3, we find that some latter model variants exhibit superior performance over the pure dense retrieval-based external expansion model. The complementary effect of the contextualised matching models and the statistical information-based matching models explains this observation, which is also observed in other recent work (Arabzadeh et al., 2021, Gao et al., 2020).

In addition, we further conduct the zero-shot evaluation of the external expansion dense models on four BEIR benchmarks and show the results in Table 5.5. Firstly, from the top half of Table 5.5, we find that performing query reformulation using ColBERT-PRF on the target dataset can improve the retrieval effectiveness on all compared datasets except DBPedia. In addition, performing the external dense expansion using ColBERT can further bring significant improvements in terms of nDCG@10 performance on both NFCorpus and Touché-2020 datasets. The ineffectiveness of dense external expansion on the TREC-COVID and NFCorpus datasets are probably due to the external corpora employed (MSMARCO), which contains less biomedical related content. Indeed, TREC-COVID and NFCorpus are biomedical corpora (see Table 2.4), while MSMARCO is a more general corpus. Moreover, MSMARCO predates the COVID-19 pandemic, and hence is not a good source of external expansion for the TREC-COVID corpus.

In response to RQ5.2(a), we find that external feedback documents obtained using dense retrieval are beneficial for both external expansion for both sparse & dense feedback and retrieval models. Applying external expansion using a dense retrieval model can significantly improve over the dense and sparse PRF models, i.e. the (a), (b) and (d) baselines in Table 5.4.

To visualise the impact of the external expansion, Table 5.6 lists three example queries from the Robust04 title and description query sets. For each query example, we show both the sparse expansion tokens generated by RM3 and the the most likely expansion tokens of the expansion embeddings selected by the ColBERT-PRF model in the target corpora and the external corpora (The FAISS ANN index is used to map an embedding back to the most likely token)[7]. The colour of the token indicates the usefulness of the expansion token, with darker indicating higher utility. Usefulness is measured by the difference in Average Precision when that expansion token is removed. From the table, it can be observed that expansion using the external corpus can produce some more useful expansion tokens than the target corpora. For instance, for the query: 'how are oscar winners selected', expansion embeddings close to the embedding of 'glamour', 'voting' and 'actors' are selected which can be seen to better identify relevant documents than the target collection (cf. 'saturday', 'don'). Later, in Section 5.5.5, we examine the extent to these expansion embeddings match exactly or inexactly with tokens in the documents (i.e. *semantic matches*).

[7] Note that this mapping from embedding to BERT WordPiece token is inexact, hence some apparently meaningless tokens, such as ##up, could actually be a useful expansion *embedding* for retrieval.

Table 5.4: External expansion for multiple representation dense retrieval. The top half table presents the results for Robust04 query sets and the bottom half table presents the results for WT10G query sets. Superscripts a-f denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model(s). The highest value for a query set is boldfaced.

	1st $\overset{R}{»}$	PRF $\overset{Q}{»}$	2nd	Robust04 (T)					Robust04 (D)				
				MAP	nDCG@10	nDCG@20	MRR	Recall	MAP	nDCG@10	nDCG@20	MRR	Recall
				Baseline Runs									
(a)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$	.275	.447	.429	.641	.738	.277	.444	.422	.625	.738
(b)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$	.284	.462	.443	.654	**.752**	.277	.468	.440	.689	**.756**
(c)	ColBERT$_{target}$	-	-	.237	.447	.421	.701	.608	.220	.435	.401	.685	.605
(d)	ColBERT$_{target}$	ColBERT-PRF$_{target}$	ColBERT$_{target}$	.273	.467	.450	.684	.677	.265	.461	.437	.668	.672
(e)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$»ColBERT	.261	.463	.442	.714	.741	.257	.460	.424	.706	.741
(f)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$ »ColBERT	.260	.458	.437	.716	.755	.254	.458	.425	.709	.749
				External Expansion: Dense External Dense Retrieval (Ours)									
	ColBERT$_{ext}$	ColBERT-PRF$_{ext}$	ColBERT$_{target}$	**.287**^{cd}	**.477**^{ac}	**.467**^{cd}	**.706**^{d}	.714^{cd}	**.281**^{abcd}	**.486**^{c}	**.459**^{c}	**.708**^{a}	.705^{cd}
				External Expansion: Sparse External Dense Retrieval (Ours)									
	BM25$_{ext}$	ColBERT-PRF$_{ext}$	ColBERT$_{target}$	.241	.429	.411	.634	.669^{c}	.231	.424	.397	.626	.644^{c}
	DPH$_{ext}$	ColBERT-PRF$_{ext}$	ColBERT$_{target}$	.243	.431	.414	.677	.674^{c}	.217	.400	.374	.614	.631^{c}
				WT10G (T)					WT10G (D)				
				Baseline Runs									
(a)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$	.202	.328	.326	.505	.711	.218	.379	.358	.584	**.751**
(b)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$	**.235**	.364	.364	.574	**.752**	**.230**	.369	.358	.622	.748
(c)	ColBERT$_{target}$	-	-	.160	.356	.337	.614	.510	.162	.360	.339	**.655**	.551
(d)	ColBERT$_{target}$	ColBERT-PRF$_{target}$	ColBERT$_{target}$	.183	**.397**	**.372**	.600	.547	.190	.393	.363	.601	.516
(e)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$»ColBERT	.199	.377	.358	.605	.711	.204	.388	.364	.692	.751
(f)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$ »ColBERT	.202	.373	.355	.605	.757	.204	.389	.363	.692	.748
				External Expansion: Dense External Dense Retrieval (Ours)									
	ColBERT$_{ext}$	ColBERT-PRF$_{ext}$	ColBERT$_{target}$	.216^{cd}	**.397**^{ac}	**.372**^{ac}	**.651**^{a}	.614^{cd}	.226^{cd}	**.408**^{ac}	**.394**^{c}	.651	.644^{cd}
				External Expansion: Sparse External Dense Retrieval (Ours)									
	BM25$_{ext}$	ColBERT-PRF$_{ext}$	ColBERT$_{target}$	.177	352	.337	.573	.590^{cd}	.199	.366	.360	.603	.645^{cd}
	DPH$_{ext}$	ColBERT-PRF$_{ext}$	ColBERT$_{target}$	.178	.360	.348	.568	.594^{cd}	.183	.345	.337	.605	.615^{cd}

5.5.2.2 RQ5.2(b): Dense Expansion on Single Representation Dense Retrieval

We now analyse external dense expansion where the pseudo-relevance feedback documents are produced by the ANCE model, then provided as input for the ANCE-PRF to refine the query representation. Table 5.7 presents the performance of these configurations, as well as the baselines. Firstly, we see that among the dense retrieval models, performing ANCE-PRF on both Robust04 and WT10G target collections improves over the zero-shot ANCE dense retrieval, i.e. baseline 'd' outperforms baseline 'c' in both collections in Table 5.7). We also observe that the external dense retrieval model achieves the highest performance among the three dense retrieval models on all metrics and significantly improve over ANCE performed only on the target collection.

When comparing to the baselines, we notice that the performance of all the zero-shot dense retrieval models on all query sets is lower than sparse expansion models. However, refining the query representation using the pseudo-relevance feedback information of the local collection helps to improve the zero-shot retrieval performance. Performing ANCE-PRF augmentation using the pseudo-relevance feedback documents from a high-quality external corpus results in further improvement. This indicates that the refined query representation from external dense retrieval encapsulates more broad knowledge from the external collection to represent the query. Moreover, compared with the sparse expansion models followed by the ANCE reranker baselines, we notice

Table 5.5: Zero-shot performance in terms of nDCG@10 on BEIR (Thakur et al., 2021). Superscripts a and b denote significant improvements (paired t-test with Holm-Bonferroni correction,p < 0.05) over the indicated baseline model(s). The highest nDCG@10 score on a given dataset is boldfaced. W/L denotes the number of queries of our Dense External Expansion models improved/degraded in terms of the nDCG@10 score of the ColBERT or ANCE model on a given dataset.

Dataset	Normal	Target PRF		External Expansion	
	(a) ColBERT_{target}	(b) $\text{ColBERT}_{target} \overset{R}{\gg} \text{ColBERT-PRF}_{target} \overset{Q}{\gg} \text{ColBERT}_{target}$	(W/L)	$\text{ColBERT}_{ext} \overset{R}{\gg} \text{ColBERT-PRF}_{ext} \overset{Q}{\gg} \text{ColBERT}_{target}$ (ours)	(W/L)
DBPedia	**.392**	.387	(121/202)	.353	(154/180)
NFCorpus	.316	.321	(116/87)	**.332**^{ab}	(119/81)
T-COVID	.533	**.548**	(26/18)	.507	(24/23)
Touché-2020	.307	.348	(33/13)	**.353**^{a}	(32/16)
	(a) ANCE_{target}	(b) $\text{ANCE}_{target} \overset{R}{\gg} \text{ANCE-PRF}_{target} \overset{Q}{\gg} \text{ANCE}_{target}$	(W/L)	$\text{ANCE}_{ext} \overset{R}{\gg} \text{ANCE-PRF}_{ext} \overset{Q}{\gg} \text{ANCE}_{target}$ (ours)	(W/L)
DBPedia	.265	.268	(132/137)	**.292**^{ab}	(179/106)
NFCorpus	.236	.239	(86/83)	**.258**^{ab}	(104/63)
T-COVID	.392	**.430**	(28/18)	**.430**	(27/19)
Touché-2020	.291	.292	(27/18)	**.296**	(27/18)

that applying the ANCE reranker degrades the performance in terms of MAP, nDCG@10 and nDCG@20. This indicates that there is still a large gap in performing the semantic search based on the single representation of the lexical matching.

Moreover, Table 5.5 presents the zero-shot performance evaluation of dense expansion on a single representation dense retrieval model on BEIR benchmarks. From the bottom part of Table 5.5, we find that dense external expansion using ANCE-PRF exhibits the highest nDCG@10 performance on all the four compared datasets and significantly outperform both the ColBERT and ColBERT-PRF models that are performed entirely on the target datasets. This indicates the usefulness of the external expansion for effective zero-shot evaluation on different benchmarks. This is because ANCE-PRF uses a supervised way of implementing the pseudo-relevance feedback and building upon the large pre-trained BERT model. Thus, the large pre-trained BERT model is capable of generating medical-related knowledge while performing the query refinement even without relevant PRF information as input. While highly effective for ColBERT-PRF, as it performs the PRF technique in an unsupervised way, it might be sensitive to the quality of the external corpus. If there is no relevant context provided by the external corpus, ColBERT-PRF can not create the relevant expansion embeddings out of thin air.

Overall, in response to RQ5.2(b), we find that external dense expansion on single representation dense retrieval helps to improve zero-shot dense retrieval performance for both Robust04 and WT10G.

5.5.3 RQ5.3: Sparse-obtained External Feedback for Dense Retrieval

Besides the dense pseudo-relevance feedback retrieval based on the dense external retrieval as the first stage, we further investigate the performance of the external expansion on the sparse retrieval scenarios. More specifically, we study two external sparse retrieval models using BM25

Table 5.6: Qualitative analysis: Examples of the expansion tokens generated by the sparse, namely RM3, and dense PRF, namely ColBERT-PRF, models on the target collection and the external collection for the Robust04 title and description query sets. Expansion tokens (the selected expansion tokens for RM3, or the most likely token for a given expansion embedding for ColBERT-PRF) with higher usefulness are highlighted in a darker colour.

Robust04 title queries	
Original query terms	308: implant dentistry
Sparse expansion terms (Target)	colleg chiropract implant prosthesi dentistri dental patient dentist devic 1987
Sparse expansion terms (External)	offer dental gener implant teeth tooth dentistri cosmet jaw whiten
Dense Expansion tokens (Target)	implant tooth settlement insurance products ##rs life million sales ##1
Dense Expansion tokens (External)	jaws dentistry titanium fuse ##´ implant dental tooth ##com replacement
Original query terms	632: southeast asia tin mining
Sparse expansion terms (Target)	burmes burma deleg mine southeast asia myanmar prime command gen
Sparse expansion terms (External)	or mine countri tin southeast china indonesia east asia hemisphe
Dense Expansion tokens (Target)	burma cart tin mining vo followed defense where general 000
Dense Expansion tokens (External)	bolivia indonesia cass ##´ tin england northern times world also
Original query terms	636: jury duty exemptions
Sparse expansion terms (Target)	serv , man , eslick , exempt , summon , duti , command , juri , murder , soldier .
Sparse expansion terms (External)	2 juri, guidelin , servic , excus , exempt , employe , receiv , duti , court , year .
Dense Expansion tokens (Target)	coating , soldier , jury , murder , fees , ##la , regulatory , city , ##2 , we .
Dense xpansion tokens (External)	summons , correspondence , ##ror , jury , ##´ , circuit , duty , bar , five , business .
Robust04 description queries	
Original query terms	633: what is the history of the welsh devolution movement
Sparse expansion terms (Target)	movement vote labour nationalist assembl northern scottish histori elect devolut
Sparse expansion terms (External)	assembl british richard govern wale welsh scottish devolut histori report
Dense Expansion tokens (Target)	wales poll ##16 38 won put against cent 000 or
Dense Expansion tokens (External)	odds dev literary ##´ scotland ##ol taken community history should
Original query terms	671: find documents that cite the specific benefits the salvation army provides those in need
Sparse expansion terms (Target)	document , find , armi , ford , chariti , benefit , salvat , bush , shop , cite .
Sparse expansion terms (External)	document , includ , contact , armi , salvat , assist , specif , benefit , nearest .
Dense Expansion tokens (Target)	northern , valley , se , support , ##ly , many , should , we , who , one .
Dense Expansion tokens (External)	salvation , accepts , rehabilitation , ##´ , documentation , ##ible , army , serving , 24 , health .
Original query terms	685: how are oscar winners selected
Sparse expansion terms (Target)	award box academi pictur select best oscar nomin film winner
sparse expansion terms (External)	select white best award gown academi actress worn oscar
Dense Expansion tokens (Target)	emmy nominations film saturday don ##40 " " has as
Dense Expansion tokens (External)	glamour winners voting oscar actors ##´ ##up film members actually

and DPH as the initial stage for both single and multiple representation dense-PRF paradigms. Table 5.4 presents the results of the sparse external dense retrieval followed by the ColBERT-PRF query expansion and ColBERT retrieval for both Robust04 and WT10G. Firstly, compared with the baseline runs, we see that using either BM25 or DPH as initial stage retrieval models lead significant improvement over ColBERT E2E and ColBERT-PRF but underperform the ColBERT-PRF model on the target on other metrics as well as other baseline runs. This demonstrates that sparse external expansion is a benefit for retrieving more relevant documents. Secondly, compared

Table 5.7: External expansion for single representation dense retrieval. The top half table presents the results for Robust04 query sets and the bottom half table presents the results for WT10G query sets. Superscripts a-f denote significant improvements (paired t-test with Holm-Bonferroni correction, $p < 0.05$) over the indicated baseline model(s). The highest value for a query set is boldfaced.

	1st $\overset{R}{\gg}$	PRF $\overset{Q}{\gg}$	2nd	Robust04 (T)					Robust04 (D)				
				MAP	nDCG@10	nDCG@20	MRR	Recall	MAP	nDCG@10	nDCG@20	MRR	Recall
				Baseline Runs									
(a)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$	.275	.447	.429	.641	.738	**.277**	.444	.422	.625	.738
(b)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$	**.284**	**.462**	**.443**	**.654**	.752	**.277**	**.468**	**.440**	.689	.756
(c)	ANCE$_{target}$	-	-	.131	.324	.295	.578	.539	.156	.369	.331	.641	.576
(d)	ANCE$_{target}$	ANCE-PRF$_{target}$	ANCE$_{target}$	.155	.345	.312	.586	.541	.165	.381	.343	.649	.563
(e)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$ »ANCE	.193	.388	.363	.646	.741	.214	.421	.386	.702	.741
(f)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$ »ANCE	.193	.384	.359	.643	**.755**	.215	.428	.391	**.712**	**.749**
				External Expansion: Dense External Dense Retrieval (Ours)									
	ANCE$_{ext}$	ANCE-PRF$_{ext}$	ANCE$_{target}$	.180^{cd}	.388^{cd}	.354^{cd}	.610^{cd}	.585^{cd}	.183	.403	.368	.665	.585
				External Expansion: Sparse External Dense Retrieval (Ours)									
	BM25$_{ext}$	ANCE-PRF$_{ext}$	ANCE$_{target}$	.178^{cd}	.388^{c}	.357^{cd}	.630	.547^{d}	.181	.399	.362	.658	.551
	DPH$_{ext}$	ANCE-PRF$_{ext}$	ANCE$_{target}$	.175^{cd}	.381^{c}	.350^{c}	.616	.550^{d}	.172	.370	.336	.618	.528
				WT10G (T)					WT10G (D)				
				Baseline Runs									
(a)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$	.202	.328	.326	.505	.711	.218	**.379**	**.358**	.584	**.751**
(b)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$	**.235**	**.364**	**.364**	**.574**	**.752**	**.230**	.369	**.358**	**.622**	.748
(c)	ANCE$_{target}$	-	-	.081	.224	.196	.452	.404	.110	.283	.256	.606	.453
(d)	ANCE$_{target}$	ANCE-PRF$_{target}$	ANCE$_{target}$	.103	.266	.240	.491	.434	.108	.289	.257	.589	.457
(e)	BM25$_{target}$	RM3$_{target}$	BM25$_{target}$ »ANCE	.172	.328	.311	.579	.711	.192	.363	.346	.640	.751
(f)	DPH$_{target}$	Bo1$_{target}$	DPH$_{target}$ »ANCE	.175	.321	.304	.568	.757	.193	.363	.352	.657	.748
				External Expansion: Dense External Dense Retrieval (Ours)									
	ANCE$_{ext}$	ANCE-PRF$_{ext}$	ANCE$_{target}$	.117^{cd}	.289^{c}	.259^{c}	.541^{cd}	.452^{c}	.131^{cd}	.314	.291^{cd}	.629	.518^{cd}
				External Expansion: Sparse External Dense Retrieval (Ours)									
	BM25$_{ext}$	ANCE-PRF$_{ext}$	ANCE$_{target}$	.123^{c}	.293^{c}	.267^{c}	.580^{c}	.427	.139^{d}	.312	.289	.571	.473
	DPH$_{ext}$	ANCE-PRF$_{ext}$	ANCE$_{target}$	.112^{c}	.276^{c}	.254^{c}	.496	.413	.124	.294	.271	.517	.468

with the dense external dense retrieval models, the sparse external retrieval model exhibits lower performance. These observations are consistent for both Robust04 and WT10G experiments. This indicates that in an end-to-end ColBERT-PRF retrieval paradigm, dense retrieval is more useful than sparse retrieval as the first stage to produce high-quality pseudo-relevance feedback documents.

Moreover, in a single-representation-based ANCE-PRF scenario, Table 5.7 shows the performance of sparse external dense retrieval models for both Robust04 and WT10G datasets. Firstly, we analyse the top-half table for Robust04. We make the following observations for both the sparse external ANCE-PRF dense retrieval models: (1) they exhibit higher performance than both ANCE (row (e)) and ANCE-PRF (row (f)) performed only on target collection; (2) they show slightly lower performance compared with ANCE reranking models in row (e) and row (f); (3) they show similar performance with the dense external dense retrieval models. On the half-bottom table, we make the following observations for the sparse external dense retrieval models as follows: (1) similar to Robust04, both models outperform the ANCE and ANCE-PRF on WT10G target collection; (2) however, they show a large drop compared with the ANCE reranking models in row (e) and (f); (3) different to the observation for Robust04, sparse external dense retrieval exhibits higher performance than dense external dense retrieval models. Based on this, we find that for the

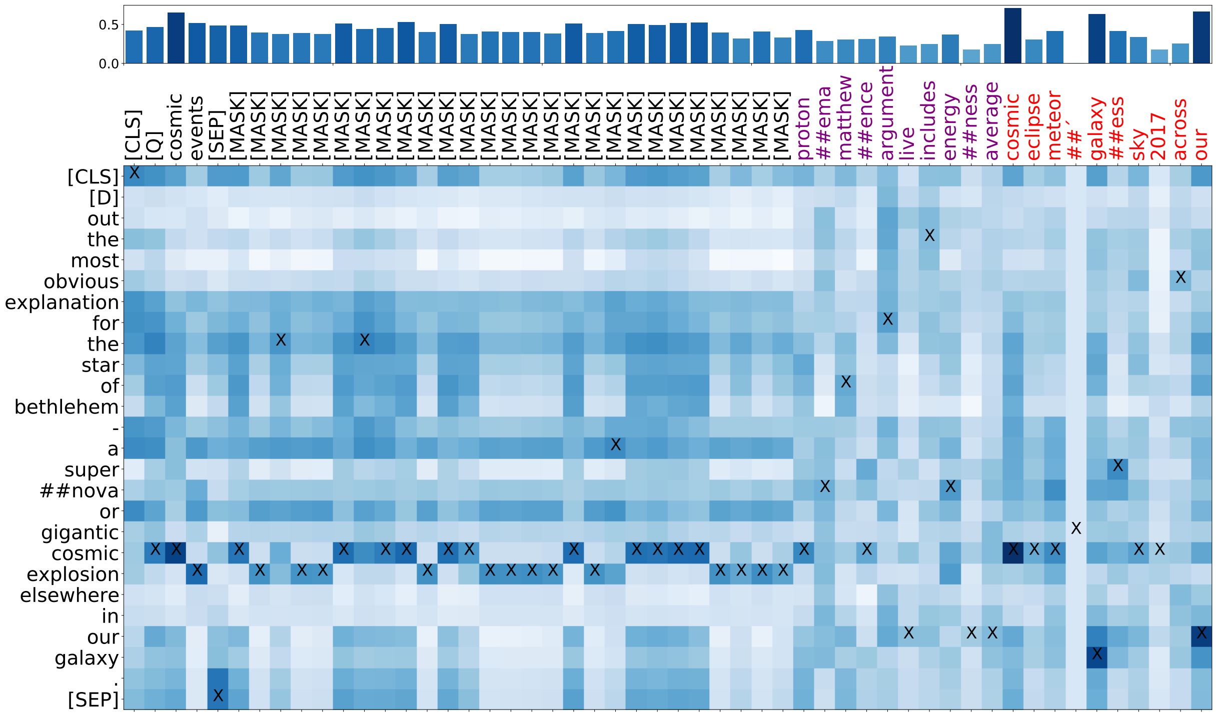


Figure 5.2: ColBERT-PRF interaction matrix between Robust04 topic (qid: 405) and document (docid: FT944-864) in an external expansion scenario. The darker shading indicates a higher similarity. The highest similarity among all the document embeddings for a given query embeddings is highlighted with a '×' symbol. The top histogram presents the magnitude of contribution for each query embedding to the final score of the document. The expansion tokens generated from the target index are highlighted in purple colour while the expansion tokens generated from the external index are highlighted in red colour.

single-representation ANCE-PRF dense retrieval model, sparse external retrieval as the first stage could also produce high-quality feedback documents to refine the query representation using the ANCE-PRF model.

Thus, in response to RQ5.3, we find that sparse external retrieval as the initial ranking stage is not sufficient to improve the performance of a multiple representation-based ColBERT-PRF dense retrieval model. However, for the single-representation ANCE-PRF dense retrieval model, sparse external first-stage retrieval can improve the retrieval performance over the ANCE baseline, although the ANCE baseline is comparatively weak (emphasising the difficulty of zero-shot single-representation dense retrieval).

5.5.4 Summary of Observations

We now report a summary of the observations from the above experiments.

Dense External Expansion, Sparse Retrieval - Section 5.5.1: We observe that external sparse expansion exhibits a similar performance to the target expansion sparse retrieval models. Moreover, external dense expansion can bring significant improvements over sparse retrieval models with expansion only performed on the target (12% improvement for Robust04 in nDCG@10: 0.432 → 0.482 and 28% for WT10G: 0.324 → 0.420 in Table 5.3).

Dense External, Dense Retrieval - Section 5.5.2 We find that external expansion using ColBERT can significantly improve over the dense retrieval models as well as the dense retrieval with target query expansion (7% improvement for Robust04 in nDCG@10: 0.447 → 0.477 and 21% 0.328 → 0.397 in Table 5.4); Similarly, external expansion using ANCE can improve the retrieval effectiveness of the dense retrieval (by 20% for Robust04 on nDCG@10: 0.324 → 0.388 and by 29% for WT10G: 0.224 → 0.289 in Table 5.7). In addition, performing dense external expansion using ColBERT or ANCE can result in further improvements on four BEIR datasets in Table 5.5.

Sparse External, Dense Retrieval - Section 5.5.3: We find that sparse external expansion brings limited useful information for ColBERT to improve the followed-up dense retrieval effectiveness, and that even applying sparse external retrieval as the initial stage can bring useful feedback documents to improve over the dense retrieval on target collection. This emphasises the continuing utility of external expansion in general, even for modern retrieval models.

Overall Performances: The highest nDCG@10 performances observed for Robust04 are 0.482 for title queries and 0.490 for description queries performing external expansion using ColBERT for sparse retrieval (see Table 5.3). The highest nDCG@10 performances for WT10G are 0.420 for title queries and 0.534 for description queries, obtained by performing external expansion using ANCE for sparse retrieval. Finally, the overall baseline dense retrieval results are not as effective as sparse retrieval in these zero-shot settings, which emphasises the overall difficulty of zero-shot dense retrieval. However, the use of dense external expansion can significantly improve effectiveness (e.g. for ColBERT-PRF, nDCG@10 0.447→0.477 in Table 5.4), it can achieve similar performance to the best sparse retrieval (e.g. 0.482 in Table 5.3 is not statistically distinguishable from 0.477). This demonstrates the benefit of external expansion for effective zero-shot dense retrieval. In the next section, we analyse to explain the effectiveness of ColBERT-PRF.

5.5.5 Semantic Matching Analysis for ColBERT-based External Expansion

We now analyse the extent to which ColBERT-PRF prefers exact matches versus inexact (semantic) matching using the Semantic Match Proportion (SMP) calculated using Equation 4.5 in Section 4.3.3.4. In this section, we further investigate the extent to which semantic matching occurs when performing query expansion using PRF documents generated from a high-quality external collection compared to only the target collection.

Figure 5.2 depicts the interaction matrix of the ColBERT-PRF model in an external expansion scenario between the Robust04 title topic: “cosmic events” and its top-ranked document. Across the top, the original query tokens, along with the ‘[MASK]’ query embeddings added for “query augmentation” (cf. Section 2.3.2) are in black; the (most likely) tokens identified by ColBERT-PRF from the target corpus are shown in purple, and those identified by ColBERT-PRF from the external corpus are in red. On analysis of the figure, we observe that for the original query token

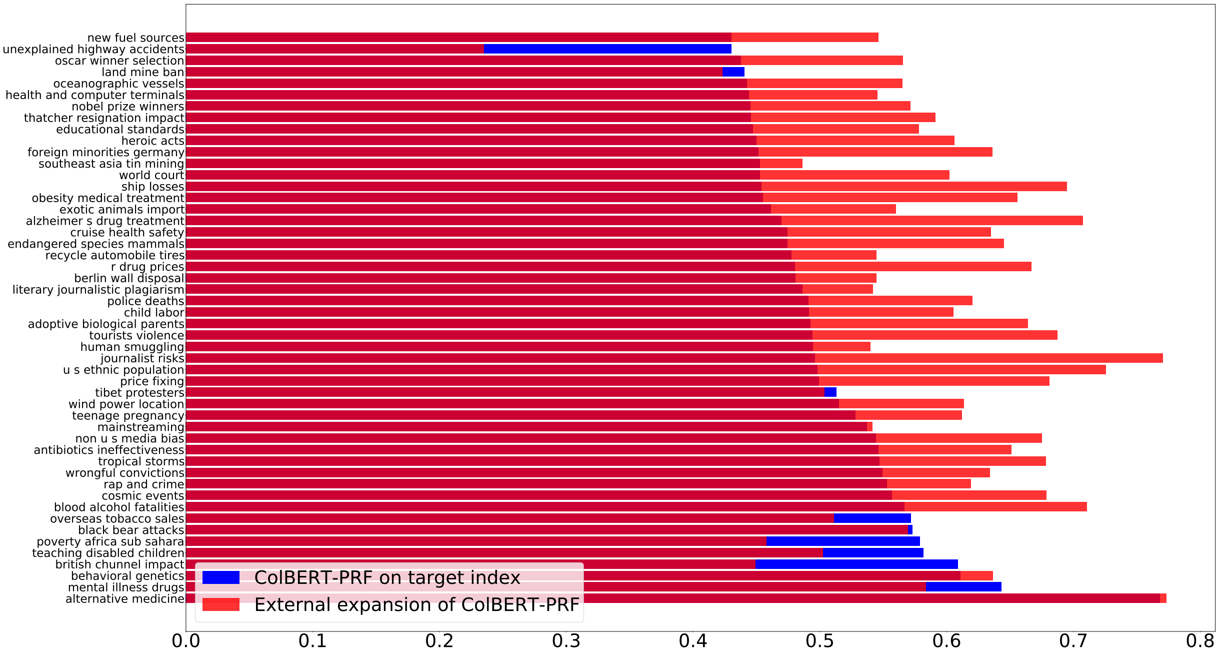
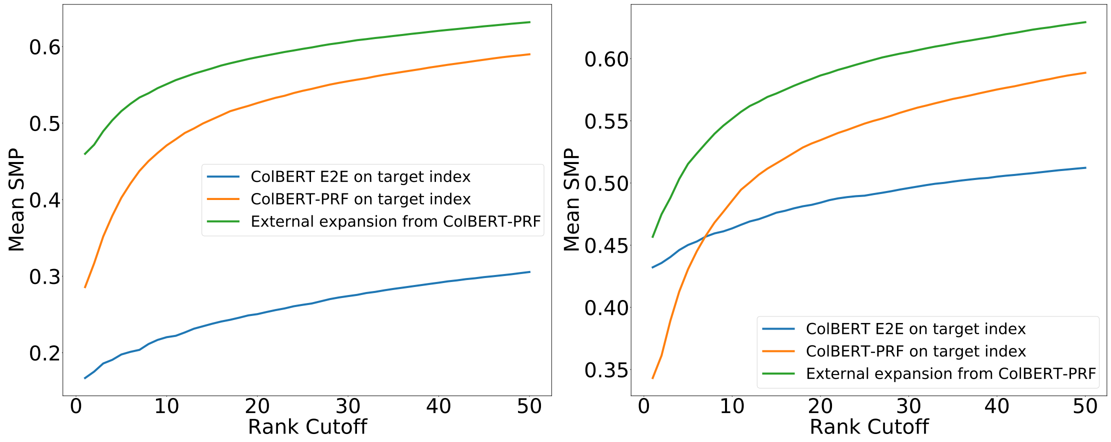


Figure 5.3: Per-query Semantic Matching Proportion for 50 of the Robust04 title topics.

'cosmic' experiences exact matching as the token is in the same form with its corresponding returned highest Max-Sim scored document token. In contrast, the original query token 'events' experiences semantic matching, as its corresponding document token with the highest Max-Sim score is 'explosion'. Indeed, this match is semantically contextualised in nature, in that it is unlikely that 'events' would somehow match 'explosion' except in the context of 'cosmic'. It is also possible to see the '[MASK]' query embeddings added by ColBERT are mostly similar in nature to the original query terms 'cosmic' and 'events', by virtue of the fact their Max-Sim matches are with the same document tokens.

When looking at the tokens for the expansion embeddings, we see that the target index generates expansion embeddings that seem unrelated to the query (e.g. 'matthew'). In contrast, the tokens for the externally sourced expansion embeddings are more related in nature to the query ('eclipse', 'meteor', 'galaxy'). Of these, 'galaxy' experiences an exact match, while other expanded embeddings experience semantic matching (e.g. 'eclipse' matches with 'cosmic').

Now, we measure the difference of the semantic matching proportion performances with and without applying the external expansion from the multiple representation dense retrieval. Figure 5.3 depicts the per-query semantic matching proportion for the first 50 Robust04 title queries against the top 1 document for the external expansion of ColBERT-PRF and ColBERT-PRF on the target index. We observe that among the sampled queries, 41/50 queries' semantic matching values are increased when performing ColBERT-PRF using feedback documents from the external collection rather than the target. This indicates that the pseudo-relevance feedback documents generated from the external collection contain broader and more useful information than the small target collection to refine the original query representation to get closer to the relevant



(a) Mean SMP on Robust04 title queries (b) Mean SMP on Robust04 description queries

Figure 5.4: Mean Semantic Matching Proportion (Mean SMP) as rank varies.

document representations in the semantic matching space.

Next, we investigate the Mean SMP of different approaches, namely the ColBERT E2E on the target index, the ColBERT-PRF on the target index and the external expansion from ColBERT-PRF, at different rank cutoffs, k. In particular, Figure 5.4 depicts the observed Mean SMP on both the Robust04 title-only and description-only query sets. On observing Figure 5.4a, we find that both of the ColBERT-PRF based approaches exhibit higher mean semantic matching proportion than the ColBERT E2E approach performed on different rank cutoffs. In addition, external expansion from ColBERT-PRF shows higher Mean SMP than ColBERT-PRF applied using only the target index. Indeed, as most of the Robust04 title-only queries are short queries, their information needs cannot be sufficiently described only using the original query representation. This observation further verifies the findings from Figure 5.3. Expanding the ColBERT query representations with expansion embeddings results in a better query representation, while expanding from a higher quality external corpus will further improve the representation. Next, from Figure 5.4b, we observe that for the description queries, at the initial ranks, ColBERT E2E exhibits higher semantic matching than the ColBERT-PRF on the target index. Put another way, ColBERT-PRF tends to experience higher exact matching on the very high ranks than ColBERT E2E. However, across the range of rank cutoff values, external expansion from ColBERT-PRF shows the highest semantic matching compared to both ColBERT-PRF and ColBERT E2E on the target collection. Overall, we find that the external expansion for multiple representation dense retrieval results in a higher semantic matching proportion than expansion only performed in the target index. This demonstrates further the value of performing pseudo-relevance feedback using ColBERT-PRF.

5.6 Conclusion

This chapter has revisited the pseudo-relevance feedback, in the form of external expansion, as applied to improve the zero-shot retrieval and addressed our proposed third hypothesis in the thesis statement in Section 1.1. In particular, our experiments employed popular dense retrieval models from both the single representation and multiple representation families to extract useful feedback documents from the high-quality external corpus (MSMARCO). More specifically, we investigated different frameworks performing external expansion for zero-shot retrieval and conducted extensive experiments on two TREC test collections (Robust04 and WT10G) and four BEIR datasets (DBPedia, NFCorpus, TREC-COVID and Touché-2020), namely (a) *dense external expansion for sparse retrieval*, (b) *dense external expansion for dense retrieval* and (c) *sparse-obtained external feedback for dense retrieval*. Overall, we found that high-quality feedback documents obtained from both multiple representation dense retrieval (cf. Table 5.4) and single representation dense retrieval (cf. Table 5.7) can significantly improve sparse retrieval on both test collections (by 12% and 28% for Robust04 and WT10G, respectively). Moreover, we observed that performing external dense expansion can significantly outperform the zero-shot dense retrieval models on target collection. In addition, we found that pseudo-relevance feedback documents produced by the sparse retrieval model are beneficial to augment query representation. Finally, we thoroughly investigated the semantic matching analysis for ColBERT-PRF and observed that performing external expansion using multiple representation dense retrieval results in higher semantic matching proportion than performing on the target.

However, one limitation of the models discussed in this chapter is that they have only been experimented with using a single external corpus. Their effectiveness with multiple external corpora for expansion has not been quantified. In addition, the proposed models are specific to the BERT model. While their retrieval effectiveness is notable, the generalisation potential of ColBERT and ColBERT-PRF to other pretrained language models remains under-explored. This raises a natural question: can ColBERT and ColBERT-PRF models generalise to other pre-trained language models? For instance, extending to the RoBERTa (Liu et al., 2020) and ALBERT (Lan et al., 2020)-based models since they employed different tokenisation techniques to BERT model, which may impact the input representation and data processing pipeline. Therefore, in the following chapter, we delve deeper into examining the effectiveness of models that go beyond BERT.

Chapter 6

From ColBERT-PRF to Col$\star$-PRF

In Chapter 4 and Chapter 5, we examined the effectiveness of our ColBERT-PRF technique for dense query expansion and dense external expansion, respectively, and validated our second and third posed hypotheses in our thesis statement (cf. Section 1.1). All of these validations have been conducted upon ColBERT (cf. Section 2.3.2), which uses the BERT PLM (cf. Section 2.2.1). Going further, this chapter further investigates our posed third hypothesis, namely that our key ColBERT-PRF model can be effectively extended to various forms of late interaction dense retrieval models.

In particular, as introduced in Section 2.3.2, ColBERT operates based on the token-level representations of query and document and consists of two stages: the Approximate Nearest Neighbour Search (ANN Search) stage and the Contextualised Late Interaction stage. Moreover, ColBERT-PRF also operates based on the token-wise dense representations of the pseudo-relevance feedback passages. Hence, on the one hand, the representation of the query and document can have a direct impact on the matching effectiveness. However, the de-facto PLM used by ColBERT is BERT while, as we introduced in Section 2.2, there are various different types of PLMs that are more extensively trained than BERT, such as ColRoBERTa and ELECTRA, and PLMs that are more lightweight compared to BERT, such as miniLM and ALBERT.

ColBERT uses a contextualised late interaction mechanism and our proposed ColBERT-PRF model uses the weighted contextualised late interaction for scoring. The nature of the matching behaviour within the late interaction mechanism includes lexical and semantic matching, introduced in Section 4.3.3.4, depends on the vocabulary. Sub-word tokenisation is the de-facto standard tokenisation approach in neural IR, due to the advantages of a limited-size vocabulary (cf. Section2.2). Tokenisation algorithms used by common contextualised models (cf. Section2.2) include WordPiece (Schuster and Nakajima, 2012), used by the BERT and ELECTRA model, Byte-Pair Encoding (BPE) (Bostrom and Durrett, 2020), used by the RoBERTa model, and SentencePiece (Kudo and Richardson, 2018), used by the ALBERT and T5 models. Different pretrained models, and their different tokenisation algorithms, lead to different embeddings in different representation spaces. In addition, the same type of pretrained model can often be instantiated in differing sizes (number of layers, etc.), where larger models can be more effective. Therefore, in this chapter, we extend ColBERT to Col⋆, instantiating the late interaction mechanism with various pretrained models using different types of tokenisation techniques. Furthermore, we generalise our proposed ColBERT-PRF technique to Col⋆-PRF.

We extensively evaluate the retrieval effectiveness of the various extended Col⋆ and Col⋆-PRF models. In addition, we are also concerned with the matching behaviour operated within the contextualised late interaction. Therefore, we conduct the semantic matching proportion analysis introduced in Section 4.3.3.4 to further explain the (weighted) contextualised late interaction experienced by Col⋆-PRF. More specifically, we examine the semantic matching proportion values overall as well as the more fine-grained matching behaviour on various salient types of token families for both Col⋆ and Col⋆-PRF models. Finally, we also quantify the contribution of different types of matching, namely lexical matching, semantic matching, and special token matching to the overall retrieval effectiveness.

In summary, this chapter makes the following contributions: we study the effectiveness of multi-representation dense retrieval with different pretrained models with different tokenisation algorithms and we observe that: (i) ColBERT and ColBERT-PRF can be generalised upon various pretrained language models as Col⋆ and Col⋆-PRF, respectively; (ii) in terms of retrieval effectiveness, we observe that applying the late interaction mechanism upon a RoBERTa model (which employs BPE tokenisation) exhibits comparable retrieval effectiveness to ColBERT; (iii) the Col⋆-PRF technique exhibits consistent improvements in retrieval effectiveness over the corresponding underlying Col⋆ model. Moreover, our extensive semantic matching proportion analysis yields the following new findings: (iv) applying the Col⋆ and Col⋆-PRF models with the BPE tokeniser is more likely to perform semantic matching than the more common ColBERT model; (v) among various salient token families, all of the (weighted) contextualised late interaction models perform semantic matching, particularly for low IDF tokens and stopwords tokens; (vi) performing only exact matching and the special token matching contribute more than only semantic matching to the overall retrieval effectiveness. These insights help explain the matching behaviour in contextualised late interaction retrieval with and without the pseudo-relevance feedback mechanism and can shed light on the more effective dense retrieval model design and retrieval.

The remainder of this chapter is organised as follows: Section 6.1 introduces our extended Col⋆ and Col⋆-PRF models. Next, we explain the semantic matching behaviour of the proposed Col⋆ and Col⋆-PRF models in Section 6.2, including the corresponding proposed research questions (Section 6.2.1) and the experiment results (Section 6.2.2). Finally, we summarise our findings and provide future work directions in Section 3.5.

Table 6.1: Tokenisation for example inputs for 3 tokenisers, corresponding to BERT, ALBERT and RoBERTa respectively.

Technique	Example 1	Example 2
Sample Text	`casualties in ww2`	`Casualties`
WordPiece	`[CLS] casualties in w ##w ##2 [SEP]`	`[CLS] casualties [SEP]`
SentencePiece	`[CLS] _casualties _in _ ww 2 [SEP]`	`[CLS] _casualties [SEP]`
BPE	`<s> Ġcasualties Ġin Ġw w 2 </s>`	`<s> Cas ual ties </s>`

6.1 Extending ColBERT-PRF to Col⋆-PRF

In this section, we first introduce the Col⋆ and Col⋆-PRF approaches in Section 6.1.1. Next, we detail the research questions and provide the results and analysis of our proposed Col⋆ and Col⋆-PRF approaches in Section 6.1.2 and Section 6.1.3, respectively.

6.1.1 Col⋆ and Col⋆-PRF

Tokenisation is an important technique to preprocess the input text before input to a contextualised language model. In particular, as transformer-based models learn representations for each unique token, a limited-size vocabulary is important. A larger vocabulary size would cause increased memory and time complexity, and difficulty in learning accurate representations for rare tokens. For these reasons, sub-word tokenisation is usually used to split the input text into small chunks of text. Thus, frequently used words are given unique IDs, while rare words will be processed into sub-words. Prevalent tokenisation techniques used by large pretrained language models include WordPiece, Byte-Pair Encoding (BPE) and SentencePiece tokenisation techniques. For instance, WordPiece is used by BERT and miniLM; BPE is used by RoBERTa and GPT models; SentencePiece is used by ALBERT and T5 models. In particular, the BPE and WordPiece tokenisation techniques merge the characters into larger tokens but control the vocabulary size using different algorithms to maximise the likelihood of the training data. In contrast, SentencePiece treats the whole sentence as one large token and learns to split it into sub-words. Table 6.1 compares the outputs of the different tokenisation approaches for the example texts "`casualties in ww2`" and "`Casualties`". Firstly, each tokeniser has its own rule to mark the beginning and end of the sentence and whether the token is sub-word token or not (`##` vs. _ vs. `Ġ`). Moreover, we see that all three compared tokenisation techniques can produce tokens of the more frequent words with their surface word form, such as `in`. However, for the rarer words (`ww2`), the various tokenisers differ in how they split these words into sub-words and encode as tokens. For instance, WordPiece and BPE produce separate the `w`, `w` and `2` in `ww2`, while SentencePiece has a token for `ww`. Notably, RoBERTa's BPE tokeniser is case-sensitive (see also Table 6.2), and while the vocabulary contains the surface form of `casulaties`, the less frequent uppercase word is broken into three sub-word tokens. This can directly impact the matching behaviour within the late interaction mechanism, as further discussed in Section 6.2.2.1.

Indeed, different tokenisers will directly affect the generated embeddings thus affecting the model performance. For instance, studies have examined different tokenisation techniques for language model pretraining (Bostrom and Durrett, 2020, Guo et al., 2021) and for low-resource language models (Rajab, 2022, Toraman et al., 2022). However, the impact of differing tokenisers for dense retrieval has not been previously investigated. Most recently, ColBERT-X (Nair et al., 2022) replaced the BERT pretrained model with the XLM-RoBERTa pretrained model when applying ColBERT for a cross-language retrieval task. However, ColBERT-X is motivated by the cross-language abilities of the XLM-RoBERTa model and made no conclusions on the effect of the different tokenisation techniques. In this work, we not only investigate the effect of the different pretrained models in ColBERT but also study the effect of using different tokenisation techniques upon English dense retrieval. In addition, we further inspect their impact on the contextualised matching pattern occurring in the dense retrieval models.

More specifically, the characteristics of the Col$\star$ models we introduce are summarised in Table 6.2. The models can be classified within three families, according to the tokenisation technique each model uses, namely WordPiece, BPE and SentencePiece. From the table, we can see that the different base models have different vocabulary sizes and the number of parameters. Moreover, their corresponding ColBERT-like dense indices vary considerably in size.

Table 6.2: Characteristics for different Col$\star$ models with contextualised late interaction.

Col$\star$ Model	Tokeniser	Vocab. Size	Index size	Embedding Dim.	Number of Parameters	HF Base Model
ColBERT-Base	WordPiece	30,522	193G	128	1095M	`bert-base-uncased`
ColBERT-Large	WordPiece	30,522	-	128	3353M	`bert-large-uncased`
ColBERT-Tiny	WordPiece	30,522	-	128	44M	`bert-tiny`
ColBERT-Mini	WordPiece	30,522	-	128	112M	`bert-mini`
ColBERT-Small	WordPiece	30,522	-	128	288M	`bert-small`
ColBERT-Medium	WordPiece	30,522	-	128	414M	`bert-medium`
ColELECTRA-Base	WordPiece	30,522	-	128	1090M	`electra-small-discriminator`
ColminiLM	WordPiece	30,522	64G	32	227M	-
ColRoBERTa-Base	BPE	50,267	356G	128	1247M	`roberta-base`
ColRoBERTa-Large	BPE	50,267	-	128	3555M	`roberta-large`
ColALBERT-Base	SentencePiece	30,002	199G	128	119M	`albert-base-v2`
ColALBERT-Large	SentencePiece	30,002	-	128	218M	`albert-large-v2`
ColALBERT-XLarge	SentencePiece	30,002	-	128	631M	`albert-xlarge-v2`
ColALBERT-XXLarge	SentencePiece	30,002	-	128	2275M	`albert-xxlarge-v2`

For the models with the WordPiece tokeniser, we apply the late interaction mechanism upon six BERT models with various sized pretrained models, from BERT-Tiny to BERT-Large. The aim of training these variants is to investigate the impact of the number of parameters of the base model that ColBERT encoders are initialised from. In addition, for WordPiece tokeniser models, we also apply ColminiLM and ColELECTRA models. miniLM (Wang et al., 2020a) is a distilled variant of BERT, which aims to reduce the huge number of parameters while retaining BERT's performance. In our work, we use miniLM as a base model for the late interaction dense retrieval mechanism and use $m = 32$ component embeddings. This thus represents a ColBERT-like setting with minimal time- and space-efficiency overheads (Bergum, 2021). We denote this as ColminiLM. Moreover, ELECTRA has been shown to achieve higher performance than a similar-sized BERT on certain NLP tasks and can be implemented as an effective cross-encoder

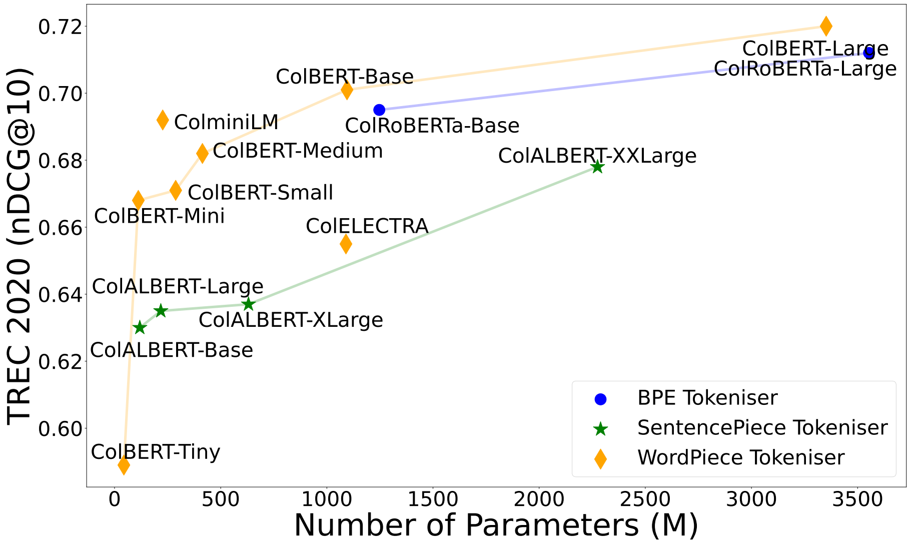


Figure 6.1: The retrieval effectiveness (y-axis: nDCG@10) of Col⋆ models on TREC 2020 query set. The x-axis shows the number of parameters of the Col⋆ models. Different markers indicate the tokenisation technique used by the Col⋆ models.

for reranking (Gospodinov et al., 2023, MacAvaney et al., 2022), but its performance has yet to be ascertained for dense retrieval. We implement the late interaction based on ELECTRA and denote this as ColELECTRA.

Secondly, to consider the BPE tokeniser, we train ColRoBERTa with both Base and Large sizes. RoBERTa employs the same model architecture as BERT but exploits the BPE tokeniser, with an increased vocabulary size wrt. ColBERT and ColminiLM. We note that RoBERTa is used as the base model for the ANCE dense retrieval model. We extend the ColBERT model using the RoBERTa base model within its BPE tokeniser, denoted as ColRoBERTa. Similar to miniLM, ALBERT aims at reducing the number of parameters of BERT by sharing parameters across transformer layers. In our experiments, we train four ColALBERT models by fine-tuning various sized base models, including ‘Base’, ‘Large’, ’XLarge’ and ‘XXLarge’ ALBERT models. ColALBERT models employ the SentencePiece tokeniser, which allows us a third tokeniser setting.

All the Col⋆ models listed in Table 6.2 are trained following the original ColBERT training setup, with a batch size of 32 and the query length and document lengths are set as 32 and 180, respectively. Table 6.2 also provides salient details and statistics of the models and their corresponding indices. In addition, for all the Col⋆ models, except ColminiLM, we fine-tune the models upto 300k iterations, selecting the final model based on reranking effectiveness on the 2019 queries. For ColminiLM, we use the checkpoint `vespa-engine/col-minilm` provided by the author of (Bergum, 2021) which was trained similarly. Since using the MSMARCO Dev query set for validation is computationally expensive, we used a smaller set of TREC 2019 queries for validation instead. All the Col⋆ models are trained with the cosine similarity method.

Figure 6.1 shows the number of parameters and the tokeniser's impact on the retrieval effectiveness of various Col⋆ models. An ANOVA study indicates that both the number of parameters and the type of tokeniser used have a significant impact on the nDCG@10 scores, at a significance level of $p < 0.05$. The performance of the models on natural language understanding tasks tends to improve with an increase in the number of trainable parameters (Kaplan et al., 2020), although this is not always the case (Zhong et al., 2021). Our findings, as displayed in Figure 6.1, indicate that for BERT-based, ALBERT-based and RoBERTa-based Col$\star$ models, retrieval effectiveness tends to increase with an increase in the number of parameters. It should be noted that larger parameterised models may be more prone to overfitting and require more computational resources for both training and inference. Additionally, the quality of the training data and the model architecture can also impact the retrieval performance of Col$\star$ models. More importantly, considering the environmentally friendly information retrieval (Scells et al., 2022), we focus on the Col$\star$ models with different tokenisation techniques and investigate the impact of the tokenisation techniques on semantic matching behaviour. To this end, we select to index ColBERT-Base, ColRoBERTa-Base and ColALBERT-Base models. We also compare with the ColminiLM model, which reduces the embedding dimension from 128 to 32.

6.1.2 Research Questions

We pose three research questions about the effectiveness of Col$\star$ and Col$\star$-PRF models, as follows:

Firstly, we investigate the effectiveness of Col$\star$ models, which serving as the initial retrieval stage of Col$\star$-PRF, by posing our first research question:

RQ6.1: How does the retrieval effectiveness vary across different contextualised late interaction models?

In addition, we investigate the effectiveness of implementing ColBERT-PRF technique across various Col$\star$ models, i.e., Col$\star$-PRF models, by posing the second research question:

RQ6.2: How does Col$\star$-PRF compare to its corresponding underlying Col$\star$ model?

Furthermore, we examine the impact of the source pseudo-relevance feedback text by controlling the initial retrieval stage of Col$\star$-PRF models, and therefore ask:

RQ6.3: What is the impact of the controlling first stage retrieval for the Col$\star$-PRF model?

6.1.3 Results and Analysis

We now present the results and analysis to address research questions RQ6.1 - RQ6.3 from Section 6.1.3.1 to Section 6.1.3.3.

6.1.3.1 RQ6.1 - Retrieval Effectiveness across Col$\star$?

To understand if the de-facto BERT base model can be replaced for implementing the late interaction mechanism, we deploy the late interaction technique on various contextualised pretrained language models (which also use varying tokenisers). Table 6.3 reports the evaluation

Table 6.3: Performance of contextualised late interaction models. The † (◇) symbol denotes statistically significant differences compared to BM25 (ColBERT). The highest value in each column is boldfaced.

Models	TREC DL 2019					TREC DL 2020				
	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP
BM25 (PyTerrier)	0.286	0.480	0.640	0.755	-	0.293	0.494	0.615	0.807	-
BM25 » Late Interaction										
ColBERT	**0.459**†	**0.713**†	0.847†	0.755	0.375	**0.484**†	**0.707**†	0.835†	0.807	0.387
ColminiLM	0.431†	0.654†◇	0.811†	0.755	0.362	0.458†	0.685†	**0.866**†	0.807	0.363
ColRoBERTa	0.458†	0.695†	**0.865**†	0.755	**0.599**	0.462†	0.695†	0.844†	0.807	**0.607**
ColALBERT	0.412†◇	0.634†◇	0.821†	0.755	0.367	0.401†◇	0.630†◇	0.751†	0.807	0.390
ANN Search » Late Interaction										
ColBERT	**0.445**†	**0.708**†	0.857†	**0.773**	0.390	**0.473**†	**0.690**†	0.832†	**0.806**	0.406
ColminiLM	0.388†◇	0.631†◇	0.811†	0.698◇	0.382	0.434†◇	0.672†	**0.860**†	0.762◇	0.388
ColRoBERTa	0.426†	0.684†	**0.866**†	0.738	**0.610**	0.423†◇	0.666†	0.828†	0.760	**0.622**
ColALBERT	0.356◇	0.613†◇	0.769	0.772	0.381	0.367†◇	0.604†◇	0.745†	0.792	0.413

results for the selected Col⋆ models for both the reranking and end-to-end dense retrieval scenarios on both TREC DL 2019 and 2020 query sets.

First, we analyse the ColminiLM model, which exploits a lightweight BERT model and uses the identical WordPiece tokeniser as ColBERT. From the reranking results in Table 6.3, we see that ColminiLM significantly outperforms BM25 and shows comparable performance to ColBERT across the measures on both TREC 2019 and 2020 query sets, except markedly lower than ColBERT in terms of nDCG@10 on TREC 2019 queries. Similarly, for the end-to-end retrieval experiments, ColminiLM exhibits significant improvements over BM25. However, compared to ColBERT, ColminiLM shows significantly lower MAP, nDCG@10 and Recall on TREC 2019 and significantly lower MAP and Recall on TREC 2020 queries. The lower performance of ColminiLM can be explained in that, as shown in Table 6.2, it requires much fewer parameters (only 20% of the ColBERT parameters). ColminiLM remains promising as it shows comparable nDCG@10 performance on the test queries (TREC 2020) and it has a smaller index size ($\sim$17% of the ColBERT index size).

Next, we analyse ColRoBERTa. we observe that ColRoBERTa exhibits comparable retrieval effectiveness to ColBERT and markedly improvements over BM25 when employed as a reranker on top of the BM25 sparse retrieval across all the reported measures. In addition, it shows comparable performance wrt. ColBERT in the dense end-to-end retrieval scenario on TREC 2019 and 2020 queries, except MAP on TREC 2020 query set. Overall, we find that ColRoBERTa is a good replacement for ColBERT.

For ColALBERT, we observe that it shows lower performance than ColBERT across all the reported measures on both reranking and end-to-end dense retrieval implementations on both query sets. Similar to ColminiLM, ColALBERT has significantly fewer parameters and a simplified model structure than ColBERT. Overall, ColALBERT has low performance in terms of the precision measures: MAP, nDCG@10 and MRR@10, and surprisingly high performance in terms of the Recall@1k.

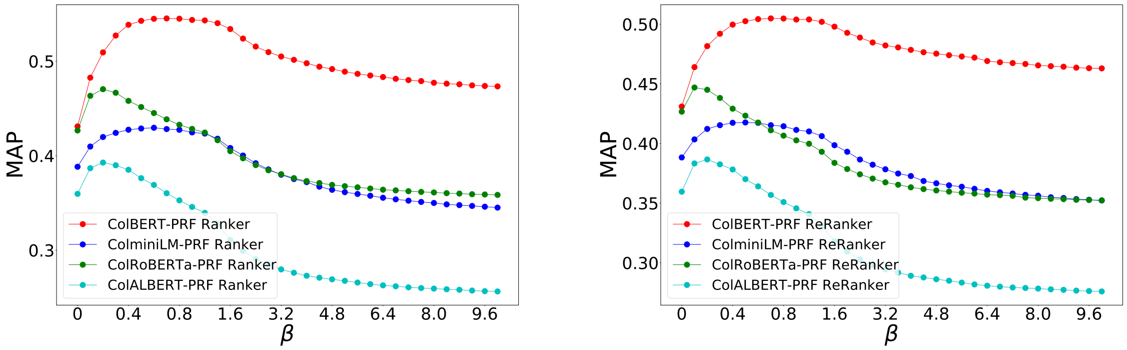


Figure 6.2: Impact of the expansion embedding weight β of Col⋆-PRF variants on the TREC 2019 query set.

Finally, it is notable that, at least on this query set, the other model families consistently do not outperform the BERT family. This suggests that more recent families of pretrained language models (ALBERT, RoBERTa) have not equated to improvements in a downstream retrieval task compared to the original BERT model.

Answer to RQ6.1: We conclude that we can implement the contextualised late interaction mechanism upon various pretrained models. More specifically, we find that, when compared to the ColBERT model, ColRoBERTa exhibits a competitive performance to ColBERT. However, consistent with the findings from Figure 6.1, we find that the ColminiLM and ColALBERT models show slightly lower retrieval effectiveness than ColBERT due to their lightweight model structures. Notably, no model family exceeds BERT in terms of effectiveness for a comparable number of parameters.

6.1.3.2 RQ6.2 - Retrieval Effectiveness for Col⋆-PRF?

We now evaluate the retrieval effectiveness of implementing the ColBERT-PRF technique, proposed in Chapter 4, on the Col⋆ models. These models have been deployed using various pretrained models and tokenisers, introduced in Section 6.1.3.1. It is important to recall that in Section 4.2.3, we introduced a parameter β to control the contribution of the expansion embeddings to the overall relevance score between an input query and a document. In particular, based on the hyperparameter study conducted in Section 4.3.3.3, $\beta = 1$ is set as the default setting for the ColBERT-PRF technique. For the Col⋆-PRF variants, namely ColBERT-PRF, ColRoBERTa-PRF, ColALBERT-PRF, and ColminiLM-PRF models, we further tune the parameter β (ranging from 0 to 10) using the validation query set, specifically the TREC 2019 passage queries. Figure 6.2 presents the impact of the expansion embedding weight β of the established Col⋆-PRF variants in both ranking and reranking scenarios of ColBERT-PRF (cf. Section 4.2.3).

From Figure 6.2, firstly, we observe that the impact of β on the Ranker and ReRanker settings for a particular Col⋆-PRF model is similar. The Ranker and ReRanker scenarios for Col⋆-PRF models

Table 6.4: Performance of Col⋆-PRF models. The † symbol denotes statistically significant differences compared to the corresponding Col⋆ E2E model. The highest value in each column is boldfaced.

Models	**TREC DL 2019**					**TREC DL 2020**				
	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP
ColBERT E2E	0.445	0.708	0.857	0.773	0.390	0.473	0.690	0.832	0.806	0.406
ColBERT-PRF Ranker	**0.532**†$^{+19.3\%}$	**0.731**$^{+3.2\%}$	0.853	**0.866**†	0.409	**0.495**†$^{+4.7\%}$	**0.714**$^{+3.5\%}$	0.825	**0.855**†	0.456
ColBERT-PRF ReRanker	**0.481**†$^{+8.1\%}$	**0.731**$^{+3.2\%}$	0.857	0.773	0.411	**0.494**$^{+4.4\%}$	**0.719**$^{+4.2\%}$	0.849	0.806	0.457
ColminiLM E2E	0.388	0.631	0.811	0.698	0.382	0.434	0.672	**0.860**	0.762	0.388
ColminiLM-PRF Ranker	0.423†$^{+9.4\%}$	0.657$^{+4.1\%}$	0.788	0.761†	0.454	0.451†$^{+3.9\%}$	0.681$^{+1.3\%}$	0.805	0.818†	0.465
ColminiLM-PRF ReRanker	0.408†$^{+5.2\%}$	0.657$^{+4.1\%}$	0.788	0.698	0.460	0.444†$^{+2.3\%}$	0.679$^{+1.1\%}$	0.807	0.762	0.470
ColRoBERTa E2E	0.426	0.684	**0.866**	0.738	**0.610**	0.423	0.666	0.828	0.760	**0.622**
ColRoBERTa-PRF Ranker	0.476†$^{+11.7\%}$	0.707$^{+3.4\%}$	0.827	0.813†	0.555	0.463$^{+9.5\%}$	0.677$^{+1.7\%}$	0.792	0.817†	0.604
ColRoBERTa-PRF ReRanker	0.456†$^{+7.1\%}$	0.707$^{+3.4\%}$	0.827	0.738	0.555	0.457†$^{+8.1\%}$	0.675$^{+1.4\%}$	0.792	0.760	0.604
ColALBERT E2E	0.356	0.613	0.769	0.772	0.390	0.367	0.604	0.745	0.792	0.413
ColALBERT-PRF Ranker	0.422†$^{+18.5\%}$	0.645$^{+5.2\%}$	0.758	0.829†	0.313	0.411†$^{+12.0\%}$	0.656$^{+8.6\%}$	0.771	0.845†	0.371
ColALBERT-PRF ReRanker	0.368†$^{+3.4\%}$	0.646$^{+5.3\%}$	0.761	0.772	0.311	0.407†$^{+10.9\%}$	0.656$^{+8.6\%}$	0.771	0.792	0.373

are originally introduced in Section 4.2.3, where Col⋆-PRF Ranker conducts the 2nd round of ANN search and late interaction scoring while the Col⋆-PRF ReRanker only performs another round of late interaction rescoring using the refined query representation. When comparing between the Col⋆-PRF models, we observe that the trend of ColBERT-PRF is similar to that of ColminiLM-PRF, and both reach their highest MAP value when β falls within the range of [0.4, 1.6]. This is expected, due to the fact that miniLM is a distilled version of the BERT pretrained language model. In contrast, ColRoBERTa-PRF, which uses the BPE tokeniser, achieves its highest MAP performance with a smaller β value of 0.2. Similarly, ColALBERT-PRF, which uses the SentencePiece tokeniser, also prefers a smaller β ($\beta = 0.2$) value for effective retrieval. This means that the expansion embeddings are less useful for ColRoBERTa and ColALBERT models than the BERT-based Col⋆ models. Therefore, we set $\beta = 0.2$ as the default value for ColRoBERTa-PRF and ColALBERT-PRF models while we use $\beta = 1$ for ColBERT-PRF and ColminiLM-PRF models. In Table 6.4, we compare the retrieval effectiveness of Col⋆-PRF model, instantiated as both Ranking and ReRanking scenarios, to the Col⋆ models on both TREC 2019 and 2020 query sets.

For ColBERT-PRF, it should be noted that, in line with (Wang et al., 2023d), we use the ColBERT model trained with a full 200k steps and a batch size of 32 rather than the one that was trained with 44k steps and used in Chapter 4. In Table 6.4, we also observe that the ColBERT-PRF models significantly outperform the ColBERT-E2E model on both TREC 2019 and 2020 query sets. These results aligh with the performance of the ColBERT-PRF with fewer training steps (44k steps) reported in Table 4.1 in Section 4.3. Additionally, for the ColminiLM-PRF models, we observe significant improvements over the ColminiLM E2E model. These observations emphasise the effectiveness of our proposed ColBERT-PRF for the BERT-based contextualised late interaction mechanism, particularly when used with the WordPiece tokeniser. When comparing the ColBERT-PRF models with the ColminiLM-PRF models, namely ColminiLM vs. ColBERT and ColminiLM-PRF vs. ColBERT-PRF for Ranker or ReRanker, we

Table 6.5: The pseudo-relevance feedback passages in the controlled Col$\star$-PRF Models.

Query	what is a active margin
1st PRF	'Best Answer: An **active margin** is a **tectonic plate boundary**, a passive margin is a compositional transition within a **tectonic plate**. As you probably know the theory of plate tectonics posits the outer layer of the Earth consist of plates which move relative to each other driven by convection in the Mantle.'
2nd PRF	'An **active plate margin** is an actual **plate boundary**, where **oceanic crust** and **continental crust crash** into each other. Active plate margins are often the site of earthquakes and volcanoes. **Oceanic crust** created by **seafloor** spreading in the East Pacific Rise, for instance, may become part of the Ring of Fire, the horseshoe-shaped pattern of **volcanoes** and **earthquake zones** around the **Pacific ocean basin**.'
3rd PRF	'**Active Margins** occur where **oceanic lithosphere** is being **subducted beneath the edge of the continent**. the result is a relatively **narrow margin** that consists of highly **deformed sediment** that was scraped from the **decending lithospheric slab and plastered against the margin of the overriding continent.** '
Query	what is the most popular food in switzerland
1st PRF	'Italian **cuisine is popular** in contemporary **Switzerland**, **particularly pasta and pizza. Foods** often associated with **Switzerland include cheese and chocolate. Swiss cheeses, in particular, Emmental cheese, Gruyere, Gruyère, vacherin And, appenzeller are Famous swiss**. **Products the most popular cheese dishes** are fondue And. racletten the Italian-speaking part of Switzerland, the Ticino area, one will find a type of **restaurant** unique to the region. The Grotto is a rustic **eatery**, offering **traditional food** ranging from **pasta to homemade meat specialities**. **Popular dishes are Luganighe and Luganighetta, a type of artisan sausages.**'
2nd PRF	'Finally, there are a lot of **sweets**, including the second type of food that **Switzerland is world famous for: Swiss chocolate**. In **Switzerland, breakfast typically includes bread, butter or margarine, marmalade or honey, maybe some cheese or cereals, plus milk, cold or hot chocolate, tea or coffee.**'
3rd PRF	'**Switzerland. The food of the Swiss is unusual in that it has so many regional influences from the cuisine of its neighbours.** This includes the French, German and Italians. Historically, Switzerland was a farming country, and the **most popular crops and foods include cheese and potatoes as well as chocolate.** The food in **Europe** can be **characterized** by four categories: meats, sugar, cereals, and fats. Meats include tripe, fish, blood sausages, and wild game. Brought from India and the New World, cane sugar became a necessary **ingredient in European recipes and foods.**'

observe that the performance of a ColminiLM model exhibits a substantial decrease compared to its corresponding ColBERT model. This aligns with the observation we made in Section 6.1.3.1: the lightweight nature of ColminiLM comes at the cost of degraded retrieval effectiveness.

For ColRoBERTa-PRF models, we observe that both ColRoBERTa-PRF Ranker and ReRanker models exhibit significant improvements over the ColRoBERTa E2E model on both compared query sets (upto 9.4% on TREC DL19 and 3.9% on TREC DL20 queries in terms of MAP@1k). In addition, ColRoBERTa-PRF Ranker leads to slight improvements over the ReRanker scenario. Similar to ColRoBERTa models, for ColALBERT-PRF models, we observe that ColALBERT-PRF Ranker model markedly outperforms ColALBERT E2E model on both query sets (upto 18.5% on TREC DL19 and 12% on TREC DL20 queries in terms of MAP@1k). These observations indicate that our proposed ColBERT-PRF technique can generalise to various Col$\star$-PRF models and is still effective across various underlying PLMs and tokenisation techniques.

Answer to RQ6.2: To summarise, we find that our proposed ColBERT-PRF generalises well to Col$\star$-PRF models with different PLMs and tokenisation techniques. More specifically, Col$\star$-PRF models lead significant improvements over the corresponding Col$\star$ models. In particular, ColALBERT-PRF brings upto 18.5% and 12% improvements over ColALBERT on the TREC 2019 & 2020 queries, respectively.

Table 6.6: Examples of the expanded queries by the controlled Col⋆-PRF models with the same first stage retrieval using ColBERT E2E on the two example queries. A token with a darker red colour indicates its higher effectiveness contribution.

Original query terms	Original query tokens	Most likely tokens for expansion embeddings
	ColBERT-PRF	
what is a active margin	what is a active margin	##riding , oceanic , volcanoes , ##cton , margin , ##hos , transition , consist , ford , ring
what is the most popular food in switzerland	what is the most popular food in switzerland	italians , rustic , switzerland , ##gan , ##ere , chocolate , ford , mar , cheese , wild
	ColBERT » ColminiLM-PRF	
what is a active margin	what is a active margin	##erved , scraped , convergence , oceanic , volcanoes , ##ud , ##pher , margin , crust , pacific
what is the most popular food in switzerland	what is the most popular food in switzerland	##champ , cricket , switzerland , ##ud , bread , italian , cheese , unique , region , includes
	ColBERT » ColRoBERTa-PRF	
what is a active margin	Ġwhat, Ġis, Ġa, Ġactive, Ġmargin	astered , heric , Ġtransitional , Ġmargin , ĠActive , oes , onic , ect , oe , ĠPacific
what is the most popular food in switzerland	Ġwhat, Ġis, Ġthe, Ġmost, Ġpopular, Ġfood, Ġin, Ġsw, itzerland	isine , ĠLug , Ã¨re , eller , ĠSwitzerland , Ġmeats , Ġchocolate , Ġmar , ue , ages
	ColBERT » ColALBERT-PRF	
what is a active margin	_what, _is, _a, _active, _margin,	_seismic , _oceanic , _phosphorus , _passive , sphere , _basin , _crash , _lit , _continent , _margin
what is the most popular food in switzerland	_what, _is, _the, _most, _popular, _food, _in,	_neighbors , cher , _phosphorus , _switzerland , _dishes , _pur , _potatoes , _regional , _italian , _cheese

6.1.3.3 RQ6.3 - Control the PRF documents Col⋆-PRF?

We note that the first stage retrieval of the retrieval pipelines reported in Table 6.4 are varied depending on different instantiations of the Col⋆-PRF models. For instance, the retrieval pipeline for ColBERT-PRF can be expressed as `ColBERT»ColBERT-PRF»ColBERT` while ColRoBERTa-PRF can be expressed as `ColRoBERTa»ColRoBERTa-PRF»ColRoBERTa`. Different first-stage retrieval results in different pseudo-relevance feedback document sets, which serve as the source information for the expansion embeddings in Col⋆-PRF models. Moreover, pseudo-relevance feedback documents with varied quality can directly influence the effectiveness of our Col⋆-PRF technique. Therefore, to eliminate the impact of the pseudo-relevance feedback documents and examine the effectiveness of the Col⋆-PRF techniques themselves, we further control the initial stage retrieval, categorised as sparse (e.g., BM25) or dense (e.g., ColBERT), across various the Col⋆-PRF models.

Table 6.7: The effectiveness of the controlled Col⋆-PRF models with the controlled pseudo-relevance feedback information. The † (‡)symbol denotes statistically significant differences compared to BM25 » ColBERT and (ColBERT E2E) model. The highest value in each column is boldfaced.

Models	TREC DL 2019					TREC DL 2020				
	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP
BM25 » ColBERT	0.459	0.713	0.847	0.755	0.375	0.484	0.707	0.835	0.807	0.387
ColBERT E2E	0.445	0.708	0.857	**0.773**	0.390	0.473	0.690	0.832	0.806	0.406
Sparse (BM25) » Col⋆-PRF										
ColBERT-PRF	**0.477**	**0.718**	**0.859**	0.755	0.415	**0.496**	**0.713**	**0.877**	0.807	0.439
ColminiLM-PRF	0.407	0.607	0.699	0.755	0.422	0.433	0.651	0.748	0.807	0.472
ColRoBERTa-PRF	0.428	0.701	0.853	0.755	0.576	0.470	0.707	0.835	0.807	0.586
ColALBERT-PRF	0.436	0.644	0.841	0.755	0.371	0.427	0.666	0.768	0.807	0.371
Dense (ColBERT) » Col⋆-PRF										
ColBERT-PRF	**0.481**†	**0.731**	0.857	0.773	0.411	**0.494**	**0.719**	0.849	0.806	0.457
ColminiLM-PRF	0.465	$_{(0.657\rightarrow)}$0.704	0.812	**0.773**	0.432	0.479	$_{(0.679\rightarrow)}$0.707	**0.859**	0.806	0.457
ColRoBERTa-PRF	0.430	$_{(0.707\rightarrow)}$0.723	0.848	**0.773**	0.563	**0.489**	$_{(0.675\rightarrow)}$**0.724**	0.857	0.806	0.593
ColALBERT-PRF	0.456	$_{(0.645\rightarrow)}$0.700	0.853	**0.773**	0.345	0.418	$_{(0.656\rightarrow)}$0.675	0.750	0.806	0.396

In particular, we present the ColBERT-produced PRF passages in Table 6.5 and the most likely tokens for expansion embeddings selected by the Col⋆-PRF techniques using the controlled ColBERT-produced PRF passages in Table 6.6. From Table 6.6, we observe that ColRoBERTa-PRF, which is applied on a model with a BPE tokeniser, and ColALBERT, which uses the SentencePiece tokeniser, tend to select whole words as the expansion embeddings. In contrast, ColBERT-PRF and ColminiLM-PRF tend to select more tokens that correspond to partial words as the expansion embeddings. For instance, for the example query `what is the most popular food in switzerland`, all four compared Col⋆-PRF methods identify `switzerland`, either in lowercase or uppercase format, as the expansion tokens. At the same time, different PRF methods identified different expansion terms, where ColBERT-PRF identifies `##gan` as the expansion token based on the feedback terms `Luganighe` or `Luganighetta` while ColRoBERTa identifies the expansion token `isine` based on the `cuisine` in the PRF documents. This indicates Col⋆-PRF techniques can be influenced by different PLMs and tokenisation techniques during the expansion embeddings selection process thus performing our controlled experiments is necessary.

In terms of the retrieval effectiveness of the controlled PRF docs for Col⋆-PRF models, Table 6.7 presents the controlled experimental results. From Table 6.7, firstly, we observe that Col⋆-PRF with BM25-produced PRF docs exhibits higher performance than both the baselines. However, ColminiLM-PRF, and ColALBERT-PRF models exhibit lower performance than both the baselines but ColRoBERTa-PRF shows comparable effectiveness to the baselines. Secondly, for the Co⋆-PRF models with ColBERT-produced PRF docs, we observe that both ColBERT-PRF and ColRoBERTa models significantly outperform both baselines. Moreover, ColminiLM-PRF and ColALBERT-PRF exhibit comparable effectiveness to the baselines.

Furthermore, when comparing between the sparse and dense first-stage retrieval, the quality of pseudo-relevance feedback documents can impact the retrieval effectiveness for Col⋆-PRF models. More specifically, we observe that all Col⋆-PRF models benefit more from the dense first-stage retrieval than from the sparse first-stage retrieval, across both MAP@1000 and nDCG@10 measures. This indicates that higher-quality PRF documents will yield greater effectiveness for ColBERT-PRF models across different PLMs and embedding spaces.

Moreover, we compare the results presented in Table 6.7, in particular, the ColBERT as the first stage models, with the results presented in Table 6.4 (the reranker instantiations). We observe that all the Col⋆-PRF models with PRF docs produced by ColBERT model exhibit higher retrieval effectiveness than PRF docs produced by corresponding Col⋆ models, for instance, `ColBERT»ColminiLM-PRF` improves over `ColminiLM»ColminiLM-PRF`, from 0.657 to 0.704. This might be expected, as the higher quality PRF docs provide a better source for expansion embeddings.

Answer to RQ6.3: To summarise, in response to RQ6.3, we find that Col⋆-PRF techniques can be influenced by the PLMs and benefit higher retrieval effectiveness from higher quality PRF documents. In addition, with the controlled PRF documents for various Col⋆-PRF models, ColBERT-PRF and ColRoBERTa-PRF perform better than ColminiLM and ColALBERT. Overall, we find that ColBERT-PRF exhibits the highest performance among the selected Col⋆-PRF models.

6.2 Semantic Matching Analysis

The improved retrieval effectiveness of Col⋆ (Section 6.1.3.1) and Col⋆-RPF models (Section 6.1.3.2) motivates us to further investigate the semantic matching behaviour to obtain more insights. Thus, to examine more deeply how the different contextualised late interaction models perform retrieval, we turn to investigate their semantic matching behaviour. In particular, we employ the semantic match proportion measure (cf. Equation (4.5)) to measure the semantic contribution to relevance scoring of the documents with contextualised late interaction models. Similar to the interaction matrix plot shown earlier in Figure 4.6, Figure 6.3 illustrates the contextualised late interaction mechanism among a query and a document for ColBERT (left) and ColRoBERTa (right) models. For every query token, on the columns, a `X` marks the matching document tokens with the highest similarity score, hence contributing to the final relevance score, as in Equation (4.5). In this case, for ColBERT, query tokens such as `the`, `w`, and `##w` exhibit exact match as with lexically identical document tokens. At the same time, semantic matching behaviour occurs for the query tokens `why` and `enter`, matching with document tokens `because` and `entered`, respectively. However, the late interaction for ColRoBERTa produces different token forms and different lexical and semantic matches with document tokens and some query tokens. Thus, we observe that the base model and the tokenisation algorithm not only affect the model size (c.f. Table 6.2), but, more importantly, they impact the way the matching between queries and documents is conducted within the late interaction mechanism.

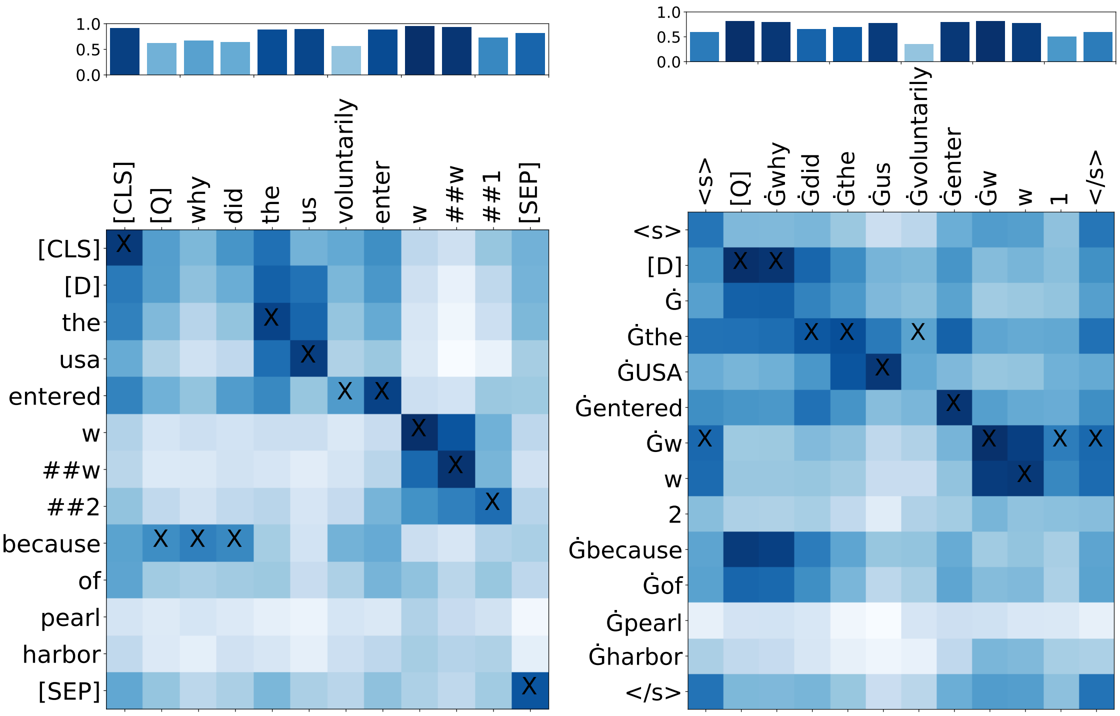


Figure 6.3: Late interaction diagrams for ColBERT and ColRoBERTa models between the query: *why did the us voluntarily enter ww1* and the document: *the usa entered ww2 because of pearl harbor*. For each column, the heatmap indicates the similarity scores among all the document embeddings for each query embedding, where the highest similarity score is highlighted with the symbol X. The top histogram depicts the magnitude of the contribution of the maximum similarity of each query embedding for the final relevance score between the query and document. The [MASK] tokens are omitted.

In the following, we further pose three research questions related to the matching behaviour of our Col⋆ models in Section 6.2.1. Then we provide the results and analysis addressing each posed research question in Section 6.2.2.

6.2.1 Research Questions

In this section, we conduct experiments to address the following research questions:
RQ6.4: How does the semantic matching behaviour vary across both Col⋆ and Col⋆-PRF models?
RQ6.5: Can we characterise the salient token families of matches, i.e., which type of tokens contribute the most to semantic matching for Col⋆ and Col⋆-PRF models?
RQ6.6: Can we quantify the contribution of different types of matching behaviour, namely the lexical match and semantic match as well as special token match, to the retrieval effectiveness of Col⋆ and Col⋆-PRF Ranker & ReRanker models?

6.2.2 Results and Analysis

As the contextualised late interaction mechanism is performed based on the token-level query and document representations, we postulate that the retrieval effectiveness of the contextualised late interaction itself, as well as the contextualised late interaction with the PRF mechanism, will be affected by the underlying tokens generated by different tokenisers. Therefore, in this section, we examine the matching behaviour for both Col⋆ and Col⋆-PRF models deployed across various types of pretrained language models and tokenisers in Section 6.2.2.1. In addition, we further investigate the semantic matching behaviour across various families of tokens in queries and documents for both Col⋆ and Col⋆-PRF models in Section 6.2.2.2. Finally, we further measure how the final retrieval effectiveness correlates with the lexical matches and the semantic matches in Section 6.2.2.3.

6.2.2.1 RQ6.4: Semantic Matching Behaviour of Col⋆-PRF

We now analyse the semantic matching proportion scores for the selected Col⋆ models reported in Table 6.3. In addition, we also analyse the semantic matching proportion scores for Col⋆-PRF models without and with controlled PRF documents presented in Table 6.4 & Table 6.7, respectively.

Firstly, for all the compared models without PRF, we report the Mean SMP values computed at rank cutoff $k = 10$. From the Mean-SMP columns in Table 6.3, we observe that ColminiLM, with the same tokenisation and vocabulary size of ColBERT, shows a similar, but slightly reduced semantic matching behaviour to ColBERT. In addition, the SentencePiece tokeniser-based ColALBERT model also shows comparable semantic matching scores. Finally, ColRoBERTa performs more of its matching in the semantic space, both for the reranking and dense retrieval scenarios. This is actually not in line with our expectations – indeed, with a larger vocabulary, we expected to see more exact matches by ColRoBERTa. We explain further the ColRoBERTa's behaviour in the next section.

We now analyse the semantic matching behaviour of Col⋆-PRF in Table 6.4. We observe that, after implementing the pseudo-relevance feedback mechanism for Col⋆ models, the semantic matching proportion values for both ColBERT-PRF and ColminiLM-PRF models slightly higher than ColBERT and ColminiLM, respectively. However, for ColRoBERTa-PRF and ColALBERT-PRF, the semantic matching proportion values are lower compared to ColRoBERTa and ColALBERT, respectively. Moreover, when comparing the Ranker and ReRanker PRF settings, we find that there is no significant difference of the retrieval effectiveness between them, as they both work on the same underlying tokeniser for a specific Col⋆-PRF model. Furthermore, the results of Col⋆-PRF models with controlled PRF documents are presented in Table 6.7, we observe that the semantic matching proportion values are similar for the Col⋆-PRF models with sparse and dense retrieval generated PRF documents.

Answer to RQ6.4: Overall, we observe that using different tokenisers, before applying PRF

Table 6.8: Salient token families of query (Q) and document (Doc) tokens.

	Notation	Type of Tokens	Example
Q	QuesToken	Question tokens	who, what, where, when, why, which, and how
Doc	SubToken	Sub-word tokens	Tokens beginning with `##` for ColBERT and ColminiLM, not beginning with space for ColRoBERTa, and not beginning with _ for ColALBERT
	SwToken	Stopwords tokens	Terrier stopwords such as `is` and `a`
	NumToken	Numeric tokens	Token corresponding to single-digit numbers
	StemToken	Stemmed tokens	Tokens in the same form as the matching query token after applying Porter stemming
	Low$_{idf}$Token	Low IDF tokens	Tokens with IDF below the 25th percentile of IDF distribution
	Med$_{idf}$Token	Medium IDF tokens	Tokens with IDF between the 25th and the 75th percentiles of IDF distribution
	High$_{idf}$Token	High IDF tokens	Tokens with IDF above the 75th percentile of IDF distribution

technique, Col$\star$ exhibits different amounts of semantic matching. In particular, the BPE tokeniser-based ColRoBERTa model exhibits a stronger preference for semantic matching compared to WordPiece and SentencePiece tokeniser-based models. Furthermore, after applying PRF, Col$\star$-PRF models experience different patterns from the Col$\star$ models in terms of the semantic matching behaviour. For instance, the semantic matching proportion values for ColRoBERTa-PRF and ColALBERT-PRF decrease from ColRoBERTa and ColALBERT models, respectively. In addition, we observe that the SMP values of ColBERT-PRF & ColminiLM-PRF models are higher than the ColBERT & ColminiLM models, respectively. Overall, we conclude that different PLMs with different tokenisers result in different inner matching behaviour for different models. Based on the findings of RQ6.4, we next inspect how semantic matching proportion values can be attributed to different families of tokens (Section 6.2.2.2), and to determine the contribution of lexical vs. semantic matching types to retrieval effectiveness (Section 6.2.2.3).

6.2.2.2 RQ6.5: SMP on Salient Token Families

We now further deepen our analysis on the internals of the late interaction mechanism with and without PRF mechanisms, by investigating the semantic matching contribution of individual query and document tokens. To this end, we identify salient families of tokens in queries and documents, based on our intuitions about how contextualised embeddings are matched. Table 6.8 summarises the identified token families. Moreover, Table 6.9 reports the observed SMP values on salient token families for Col$\star$ models, as the reranker on top of BM25 and end-to-end, on the top half and for Col$\star$-PRF models, without and with controlled the PRF docs, on the bottom half, respectively.

Analysis For Col$\star$ Models: To answer RQ6.5, we first inspect the semantic matching behaviour for different Col$\star$ models with various salient token families listed in Table 6.8. More specifically, we are more concerned about what matching behaviour is performed for the question tokens in the query and seven families of salient tokens in the document. The top half of Table 6.9 presents the semantic matching proportion scores for the above salient token families for all four

Table 6.9: Mean semantic matching proportion for the salient document token families in query and document on TREC DL 2020. The † symbol denotes improvement of the third quadrant Col⋆-PRF models to the second quadrant Col⋆ models and ‡ symbol denotes improvement of the fourth quadrant Col⋆-PRF models to the third quadrant Col⋆-PRF models. The highest value among the salient token families in each column is boldfaced.

		ColBERT	ColminiLM	ColRoBERTa	ColALBERT	ColBERT	ColminiLM	ColRoBERTa	ColALBERT
		BM25 (PyTerrier) » Late Interaction				ANN Search » Late Interaction			
Q	All Types	0.387	0.363	0.607	0.390	0.406	0.388	0.622	0.413
	QuesToken	0.085	0.087	0.090	0.067	0.087	0.089	0.091	0.070
Doc	SubToken	0.009	0.011	0.126	0.179	0.013	0.020	0.133	0.190
	SwToken	0.163	0.127	0.159	0.125	0.169	0.134	0.165	0.130
	NumToken	0.017	0.018	0.003	0.001	0.019	0.018	0.004	0.001
	StemToken	0.022	0.024	0.025	0.019	0.023	0.022	0.026	0.020
	Low$_{idf}$Token	**0.365**	**0.344**	**0.517**	**0.270**	**0.381**	**0.361**	**0.523**	**0.289**
	Med$_{idf}$Token	0.021	0.018	0.068	0.018	0.025	0.026	0.074	0.018
	High$_{idf}$Token	0.001	0.001	0.005	0.004	0.001	0.001	0.006	0.005
		Col⋆ » Col⋆-PRF				ColBERT » Col⋆-PRF			
Q	All Types	0.457†	0.470†	0.604	0.373	0.457	0.457‡	0.593	0.396‡
	QuesToken	0.011	0.010	0.037	0.031	0.011	0.010	0.037	0.043‡
Doc	SubToken	0.040†	0.042†	0.134	0.121	0.040	0.042	0.128	0.144‡
	SwToken	0.136	0.108	0.149	0.095	0.136	0.105	0.142‡	0.117‡
	NumToken	0.016	0.011	0.021†	0.007†	0.016	0.008	0.016	0.007
	StemToken	0.220†	0.037†	0.026	0.026†	0.220	0.023	0.024	0.020
	Low$_{idf}$Token	**0.406**†	**0.398**†	**0.509**	**0.285**	**0.406**	**0.400**‡	**0.500**	**0.296**‡
	Med$_{idf}$Token	0.049†	0.067†	0.075†	0.036†	0.049	0.054	0.074	0.031
	High$_{idf}$Token	0.002†	0.005†	0.010†	0.006†	0.002	0.002	0.009	0.004

contextualised late interaction models. We examine the semantic matching behaviour for both the reranking and end-to-end dense retrieval scenarios on the TREC DL 2020 query set.

From the top half of Table 6.9, we observe that question tokens occurring in the query exhibit low semantic matching scores. Among all the families of salient tokens from documents, semantic matching prefers the low IDF (i.e. frequent) tokens, followed by the family of stopwords tokens. However, semantic matching seldom occurs in the medium and high IDF tokens, which means such rare tokens are more likely to exhibit exact matching during scoring. In addition, the stemmed, numeric and sub-word token families all exhibit low semantic matching proportion. This is because the numeric tokens are usually less semantically flexible and carry more specific meaning than whole words, for instance, the ##`1` and ##`2` denotes `WW1` and `WW2`, respectively and hence do not need semantic matching. In addition, reducing a word to its base or root form in the stemmed tokens often loses the specific nuances of words, for instance, `booked` and `booking` are all stemmed as `book`. Finally, the sub-word tokens may lose the complete meaning of a word.

Furthermore, comparing the different Col⋆ models, we observe that although ColBERT, ColminiLM and ColALBERT show similar SMP values overall for all types of tokens in Table 6.3, the results in Table 6.9 indicate that their semantic matching occurs for different types of tokens. For instance, ColBERT and ColminiLM tend to perform semantic matching for the tokens with relatively low IDF scores and sub-word tokens. ColALBERT (SentencePiece) behaves more similarly to the WordPiece-based models (ColBERT & ColminiLM), except that more semantic matching comes from sub-word tokens and less from low-IDF tokens.

More interestingly, we observe that ColRoBERTa exhibits the highest semantic matching proportion scores overall in Table 6.3 and for each type of tokens in Table 6.9. In particular, it has the highest SMP value on the low IDF (i.e. frequent) tokens in Table 6.9. We explain these differences as follows: as RoBERTa's vocabulary is case-sensitive, some words can be represented by a whole token when occurring in lower-case, but resort to sub-word tokens when starting with an uppercase letter (see `Casualties` vs. `casualties` examples in the last row of Table 6.1). To make a match between these words requires a semantic match (involving relatively frequent sub-word tokens), where a case-insensitive model would have made an exact match (that would likely have been easier to learn). Indeed, the original RoBERTa authors acknowledged that their tokenisation configuration choice might not be the most effective (Liu et al., 2020). This analysis indicates the challenges for search using case-sensitive contextualised language models.

Analysis For Col⋆-PRF Models: Next, we investigate the SMP values for various token families for the Col⋆-PRF models, which are reported in the bottom half of Table 6.9. Firstly, from Table 6.9, when comparing the Col⋆ » Col⋆-PRF models (the lower left quadrant of Table 6.9) to the Col⋆ models (the upper right quadrant of Table 6.9), we observe that for the BERT-based ColBERT-PRF and ColminiLM-PRF models, with the expanded query tokens, the semantic matching proportion values increase across various types of tokens, except the subword tokens, stopword tokens and the numeric tokens, compared with the Col⋆ models. On the other hand, ColRoBERTa-PRF and ColALBERT-PRF models are more likely to perform semantic matching for numeric,as well as medium and high IDF tokens. This is because, during the expansion embedding selection, our dense query expansion technique tends to focus on tokens that have high IDF scores (cf. the model details introduced in Section 4.2.2). Overall, the low IDF tokens have the highest SMP values across all the Col⋆-PRF models. Therefore, the implementation of our dense query expansion technique can alter the matching behaviour between the query and document.

Furthermore, we examine the semantic matching with controlled PRF documents for all the Col⋆-PRF models. The results are shown in the lower right quadrant of Table 6.9. Now, we investigate the influence of the PRF documents for Col⋆-PRF models by comparing the fourth quadrant to the third quadrant of Table 6.9. We observe that ColminiLM and ColRoBERTa show slightly lower SMP values for all the types of tokens, except the low IDF tokens for ColminiLM and the stopword tokens for ColRoBERTa. ColALBERT has higher SMP values overall and on most types of tokens. These observations indicate that the quality of the pseudo-relevance feedback documents can directly impact the matching behaviours for Col⋆-PRF models.

Answer to RQ6.5: Overall, in quantifying the extent of semantic matching for various token families for both before and after ColBERT-PRF, we find that low IDF tokens are most likely to exhibit semantic matching. In addition, our ColBERT-PRF dense query expansion method can

Table 6.10: Impact of different types of matching behaviour for TREC DL 2020 on nDCG@10, and relative decrease from All (Δ). The † and ◇ symbols denote statistically significant differences compared to the BM25 and the all types matching of a model. The highest nDCG@10 value in each column is boldfaced. The ▲ (▼) symbol denotes the decrease (improvement) of the relative decrease percentage value of Col⋆-PRF models compared to the Col⋆ models.

Models	All Types	Lexical Matching		Semantic Matching		Special Token Matching	
	nDCG@10	nDCG@10	Δ	nDCG@10	Δ	nDCG@10	Δ
BM25 (PyTerrier)	0.494	-	-	-	-	-	-
BM25 (PyTerrier) » Late Interaction							
ColBERT	**0.707**†	**0.527**◇	-25.5%	0.139†◇	-80.3%	0.519◇	-26.6%
ColminiLM	0.685†	0.487◇	-28.8%	0.074†◇	-89.1%	0.523◇	-23.7%
ColRoBERTa	0.695†	0.397†◇	-42.9%	**0.261**†◇	-62.5%	**0.635**†◇	-8.6%
ColALBERT	0.630†	0.505◇	-19.8%	0.074†◇	-88.2%	0.460◇	-27.1%
ANN Search » Late Interaction							
ColBERT	**0.690**†	**0.492**◇	-28.7%	0.002†◇	-99.7%	0.384◇	-44.4%
ColminiLM	0.672†	0.426◇	-36.6%	0.001†◇	-99.9%	0.347†◇	-48.4%
ColRoBERTa	0.666†	0.350†◇	-47.5%	**0.157**†◇	-76.4%	**0.574**◇	-13.8%
ColALBERT	0.604†	0.411◇	-32.0%	0.007†◇	-98.8%	0.341†◇	-43.4%
Col⋆-PRF Ranker							
ColBERT-PRF	**0.714**†	0.568◇	-25.4%▲	0.021†◇	-97.1%▲	**0.696**◇	-2.5%▲
ColminiLM-PRF	0.681†	0.575◇	-15.6%▲	0.004†◇	-99.4%▲	0.666◇	-2.2%▲
ColRoBERTa-PRF	0.677†	0.550◇	-18.8%▲	**0.115**†◇	-83.0%▼	0.575◇	-15.1%▼
ColALBERT-PRF	0.656†	**0.594**†◇	-9.5%▲	0.008†◇	-98.8%	0.361†◇	-45.0%▼
Col⋆-PRF-ReRanker							
ColBERT-PRF	**0.714**†	0.594◇	-16.8%▲	0.029†◇	-95.9%▲	**0.696**◇	-2.5%▲
ColminiLM-PRF	0.679†	0.596◇	-12.2%▲	0.007†◇	-99.8%▲	0.664◇	-2.2%▲
ColRoBERTa-PRF	0.675†	0.550◇	-18.5%▲	**0.123**†◇	-81.8%▼	0.574◇	-15.0%▼
ColALBERT-PRF	0.656†	**0.600**†◇	-8.5%▲	0.008†◇	-98.8%	0.353†◇	-46.2%▼

alter the matching behaviour. In the next section, we conduct further experiments to quantify the contribution of different matching types to retrieval effectiveness.

6.2.2.3 RQ6.6: Contribution of Matching Types to Retrieval Effectiveness

Finally, as the ultimate outcome of matching behaviour is the ranking of the document, we analyse how the final retrieval effectiveness correlates with the lexical matches and the semantic matches for both Col⋆ and Col⋆-PRF models. To conduct this ablation, we also consider retrieval using only "special" tokens, such as `[CLS]` and `[Q]`, which always match semantically.

Analysis For Col⋆ Models: Now, we examine the retrieval effectiveness by conducting only special matching, only semantic matching, as well as special token matching (e.g., `[CLS]`, `[Q]`, `[SEP]` and `[MASK]` tokens for WordPiece tokeniser) for Col⋆ models in response to the input queries of the TREC DL 2020 query set. Table 6.10 presents the impact of performing a particular type of matching on the retrieval effectiveness (measured by nDCG@10) as well as the reduction percentage compared to all types of matching for both Col⋆ models, in the first and second groups, and Col⋆-PRF models, in the third and fourth groups in the table. From the 1st and 2nd groups of Table 6.10, we find that performing each type of matching alone results in significant reductions in effectiveness compared to all types of matching, for both the reranking and end-to-end dense retrieval scenarios. In particular, for all models except ColRoBERTa, lexical matching contributes to the highest retrieval effectiveness; for ColRoBERTa, the special tokens have excellent effectiveness (contributing 80-90% of the full effectiveness). Similarly, semantic matching alone exhibits low effectiveness but is strongest for ColRoBERTa (this again demonstrates the strong semantic properties of the ColRoBERTa token embeddings). Moreover, Table 6.10 tells us that this semantic matching is mostly concentrated on frequent (low IDF) tokens. Finally, the high performance of lexical matching is mostly related to medium and high IDF tokens - indeed, this observation echoes the finding of Formal et al. (2021b) that ColBERT is able to capture more important terms by performing exact matches. Our work systematically quantifies and generalises this finding to various contextualised late interaction models. However, different types of matching need to work together to achieve optimal retrieval effectiveness, as performing any type of matching alone will result in a significant drop in retrieval effectiveness compared to performing all types of matching.

Analysis For Col⋆-PRF Models: In addition, we further investigate the retrieval effectiveness for Col⋆ models when implementing the ColBERT-PRF technique and instantiated to both the Ranking and ReRanking scenarios, namely the Col⋆-PRF Ranker and ReRanker models. The 3rd and 4th groups of Table 6.10 report the contribution of the different types of matching to the overall retrieval effectiveness of Col⋆-PRF models, specifically the lexical matching only and semantic matching only, as well as the special token matching.

Firstly, comparing the Col⋆-PRF models, in the 3rd & 4th groups, to the Col⋆ models, the 2nd group of Table 6.10, We observe that the lexical-only type of matching contributes more to the overall nDCG@10 scores in the Col⋆-PRF models than in the Col⋆ models. For instance, performing lexical-only matching results in a decrease of 47.5% for the ColRoBERTa model, whereas it only causes decreases of 18.8% and 18.5% for the ColRoBERTa-PRF Ranker and ReRanker models, respectively. In addition, we observe that performing only semantic matching for the Col⋆-PRF models exhibit similar contributions to Col⋆ models. Finally, when comparing between the contribution of special token matching for Col⋆ and Col⋆-PRF models, we observe that performing only special token matching contributes more to the retrieval effectiveness of the BERT-based Col⋆-PRF models (i.e., ColBERT-PRF and ColiminiLM-PRF models) than the ColBERT and ColminiLM in Table 6.10, respectively. However, the importance of the special token matching for ColRoBERTa-PRF and ColALBERT-PRF is less than for the ColRoBERTa and ColALBERT models.

These observations indicate that our dense-PRF technique, typically by expanding with expansion cannot follow, one model, one tokenisertokens split by different tokenisers, alters the matching contribution to the overall retrieval effectiveness. Overall, among the three types of matching, all the Col⋆-PRF models, except ColALBERT-PRF, benefit most from special token matching, then from lexical and lowerest from semantic matching. Differently, the lexical-only matching contributes most to the ColALBERT-PRF model than the other two types of matching.

Answer to RQ6.6: In response to RQ6.6, we find that without applying ColBERT-PRF dense query expansion technique, the late interaction mechanism benefits more from lexical matching than semantic matching. In addition, special tokens, such as the `[CLS]` token, play a very important role in matching, especially for the ColRoBERTa model. However, after applying ColBERT-PRF, special token matching contributes most significantly to the overall retrieval effectiveness of the Col$\star$-PRF models. Conversely, semantic matching contributes the least to the overall retrieval effectiveness.

6.3 Conclusions

To summarise, in the proposed thesis statement in Section 1.1, we hypothesised that our key ColBERT-PRF model can be generalised to various forms of late interaction dense retrieval models. Therefore, in this chapter, we provided a comprehensive study that thoroughly investigates the semantic matching behaviour in multiple representation dense retrieval with and without the pseudo-relevance feedback mechanism. In particular, we implement the late interaction mechanism upon various contextualised pretrained models and different types of tokenisation techniques. Therefore, we extended ColBERT to Col$\star$ by applying the contextualised late interaction mechanism upon various pretrained models with different tokenisers and generalised ColBERT-PRF technique to Col$\star$-PRF techniques. We found that we can extend ColBERT-PRF model to Col$\star$-PRF models and the choice of the base pretrained model ColBERT as well as for ColBERT-PRF can greatly impact the retrieval performance, but models from the BERT family remain the most effective.

In addition, we further quantified the extent of semantic matching for dense retrieval as well as dense query expansion. Extensive experimental results yield the following findings for the Col$\star$ models without PRF techniques: (i) Col$\star$ models with different tokenisation methods show different semantic matching values, in particular, the ColRoBERTa model exhibits higher SMP values due to its case-sensitive tokeniser; (ii) among various salient families of tokens, low IDF and stopwords tokens are more likely to perform semantic matching; (iii) performing only exact matching and only special token matching contribute more than only semantic matching to all types matching retrieval effectiveness. Overall, our experimental results explain how ColBERT-like models perform retrieval, and can shed insight into more effective dense retrieval model design.

Moreover, for Col$\star$ using PRF we found that: (iv) The implementation of the ColBERT-PRF pseudo-relevance feedback technique will alter the proportion of semantic matching, for example, the SMP values for ColRoBERTa-PRF and ColALBERT-PRF may decrease compared to those of ColRoBERTa and ColALBERT models, while opposite observations can be made for ColBERT-PRF and ColminiLM-PRF models, where the SMP values may increase (cf. Table 6.4); (v) In consistent to Col$\star$ models, low IDF tokens tend to perform semantic matching in Col$\star$-PRF models, however, the probability that a type of tokens' semantic matching proportion alters depends on the implementation of the pseudo-relevance feedback technique (cf. Table 6.9); (vi) Special token matching contributes the most to the overall retrieval effectiveness of the Col⋆-PRF models. However, one limitation of the Col⋆-PRF models is that the effective dense PRF mechanism has not been transferred to the single representation dense retrieval models. We leave this as an interesting future work and provide more discussions around this in Section 8.2. Moreover, this chapter only examines the generalisation of the unsupervised manner of dense query expansion, the ColBERT-PRF model. As we have discussed in Section 2.2, the pretrained language models (PLMs), such as BERT, demonstrate a superior capability in capturing the contextual relationship between words in a sentence. BERT, in particular, provides high-quality word embeddings that effectively capture rich semantic information. However, the ColBERT-PRF model, operating in an unsupervised manner, might fall short of fully harnessing the exceptional abilities of PLMs like BERT for performing dense query expansion. In addition, ColBERT-PRF employs heuristic techniques such as clustering and IDF statistics to estimate the informativeness of the feedback embeddings. Consequently, it may have limited consideration of the surrounding context information of the feedback tokens during selection. Therefore, in Chapter 7, we investigate a supervised dense query expansion method. Specifically, we propose a BERT-based feedback weight learning model, which is trained using a contrastive weighting target for each feedback token. After training, the feedback weighting model selects and weights the usefulness of the feedback embeddings for dense query expansion.

Chapter 7

Learning Feedback Weights for Dense Query Expansion

In the previous chapters, we presented all of the essential building blocks for our thesis statement posed in Section 1.1. In particular, in Chapter 3, we proposed two possible query reformulation frameworks GenQR and GenPRF using sequence-to-sequence generative language models to generate query reformulations for sparse retrieval. Going beyond query reformulation models for effective sparse retrieval, in Chapter 4, we proposed ColBERT-PRF to generate refined query representations for improving the effectiveness of dense retrieval. Moreover, in Chapter 5, we further improved the retrieval effectiveness of ColBERT-PRF for out-of-domain performance using the external query expansion technique. Furthermore, in Chapter 6, we extended ColBERT-PRF into the Col⋆-PRF models, which are deployed across various types of underlying PLMs as well as different tokenisation techniques. These chapters well validated the posed hypotheses of our thesis statement in Section 1.1.

However, although our proposed ColBERT-PRF is effective, it performs dense query expansion in an unsupervised method, indicating that it may not fully harness the backbone BERT model for PRF. In particular, ColBERT-PRF relies on clustering and inverse document frequency (IDF) statistics for identifying the expansion embeddings — both of which are heuristics, and thus may ignore valuable context present in the embeddings. For example, for the user query *georgia run off elections*, effectiveness might be improved by adding an embedding for 'US', however, this would not likely be selected due to its low IDF (indeed, 'us' is also a pronoun, and is often included in stopword lists). Moreover, there is no direct connection between the expansion embeddings selected by the heuristic and the semantic search algorithm itself. Instead, we ask *can we train a deep language model-based feedback weighting model for identifying the discriminating expansion embeddings for query reformulation?* Therefore, in this chapter, we further investigate an effective dense expansion in a supervised manner.

Indeed, various sparse PRF models have been proposed for weighting the importance of terms occurring in the feedback documents. For instance, Clinchant and Gaussier (2011) emphasised the importance of term rarity (cf. IDF) in selecting expansion terms, a finding echoed by Roy et al. (2019) – indeed, the importance of IDF is a key insight brought into ColBERT-PRF. Going further, while there have been several approaches that have proposed supervised models for selecting high-quality expansion terms for sparse retrieval, e.g., (Cao et al., 2008, Imani et al., 2019), none of these has tackled the problem from a dense retrieval perspective.

On the other hand, in recent years, contrastive learning has been used to optimise the query and document representations produced by the BERT-based dense retrieval models in IR. More specifically, some works focus on employing various negative sampling methods, such as the in-batch (Yih et al., 2011) and cross-batch negative sampling (Qu et al., 2021), while some works mine hard negative samples for more effective dense retrieval model, for instance, in the ANCE model we introduced in Section 2.3.1. However, no effort has leveraged contrastive learning for learning the expansion weights for dense query expansion.

Based on this, in this chapter, we propose a contrastive weighting method, called CWPRF, to select and weight the usefulness of the feedback embeddings for dense expansion. More specifically, for each feedback token, we construct a contrastive objective, where, given positive and negative documents, CWPRF is trained to assign high weights to the tokens that are semantically closer to tokens occurring in the positive document than to those in the negative document. Introducing the PRF passages into the training procedure of CWPRF enables the model to take the surrounding context into account when identifying the useful tokens from the PRF passages. Meanwhile, training CWPRF with the contrastive objective allows it to learn the effective weights for expansion embeddings that are tailored for the semantic ranking task.

Overall, our contributions of this chapter are summarised as follows:

- We propose CWPRF, a contrastive weighting method for dense query expansion;
- We construct the contrastive targets and train our CWPRF model to assign high expansion weights for tokens that can discriminate the relevant documents from the non-relevant documents. Based on the predicted weights, CWPRF helps to identify useful expansion embeddings for generating refined query representations;
- We perform an extensive empirical evaluation and demonstrate how to effectively train our CWPRF in a supervised way;
- Experiments show that our CWPRF can achieve significantly higher retrieval effectiveness but with less execution time than the default ColBERT-PRF.

The remainder of this chapter is structured as follows: Section 7.1 introduces our proposed feedback weighting model; Next, Section 7.2 and Section 7.3 describes our posed research questions and the experimental setup of CWPRF, respectively. Next, we discuss the experimental results of CWPRF for each posed research question in Section 7.4. Finally, we provide concluding remarks of this chapter in Section 7.5.

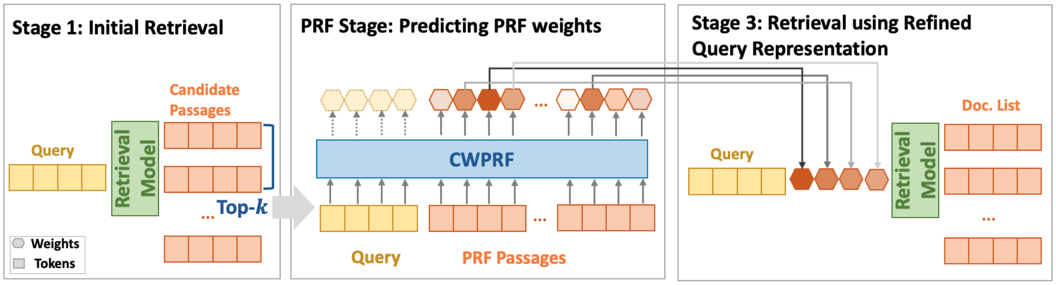

Stage 1: Initial Retrieval
Candidate Passages
Query
Retrieval Model
Top-k
Weights
Tokens
PRF Stage: Predicting PRF weights
CWPRF
Query
PRF Passages
Stage 3: Retrieval using Refined Query Representation
Doc. List
Query
Retrieval Model

Figure 7.1: Overview of CWPRF for dense query expansion.

7.1 Contrastive Weighting for Dense PRF

This section first provides an implementation overview of CWPRF for dense query expansion in Section 7.1.1. It then details the contrastive weighting method and the training procedure of CWPRF in Sections 7.1.2 & 7.1.3, respectively.

7.1.1 CWPRF Implementation Overview

An overview of CWPRF in a multiple-representation dense expansion framework is illustrated in Figure 7.1, where three stages are presented: (1) initial retrieval, (2) predicting the PRF tokens weights and (3) retrieval with the refined query representation. We note that the first and the third stages of this framework are shared with ColBERT-PRF (cf. Section 4.2).

In the initial retrieval stage, we obtain a result list in response to the original user's query q. The top f_b documents are employed as the pseudo-relevance feedback documents. Then, as input for our trained CWPRF model, we append the PRF passages to the query. The model outputs weights for each query token as well as for the feedback tokens. Finally, according to these produced weights, we identify f_e feedback tokens with high weights as our expansion tokens and append their corresponding expansion embeddings obtained from ColBERT's document encoder to the original query representation. Following conventional PRF models summarised in Section 2.4.1, going back to the Rocchio technique, the overall contribution of the expansion embeddings is further controlled by a hyper-parameter denoted by β. Finally, the refined query representation is re-issued to the underlying dense retrieval model, i.e. ColBERT, so as to return the final document list.

The core challenge, which lies in the second PRF stage, is how to accurately predict the expansion weights for the refined query representation that can more effectively perform semantic search. We propose a novel contrastive weighting model that learns to weight each feedback token individually based on the extent it will increase the score of the relevant document w.r.t. the non-relevant one(s).

Following ColBERT-PRF (Chapter 4), we again build our feedback representations from the token-level representations of the feedback documents. While we could have considered using the `CLS` of the feedback passages (which are thought to represent the meaning of the whole passage), we note that, unlike in single representation dense retrieval, the `CLS` embeddings within the token-

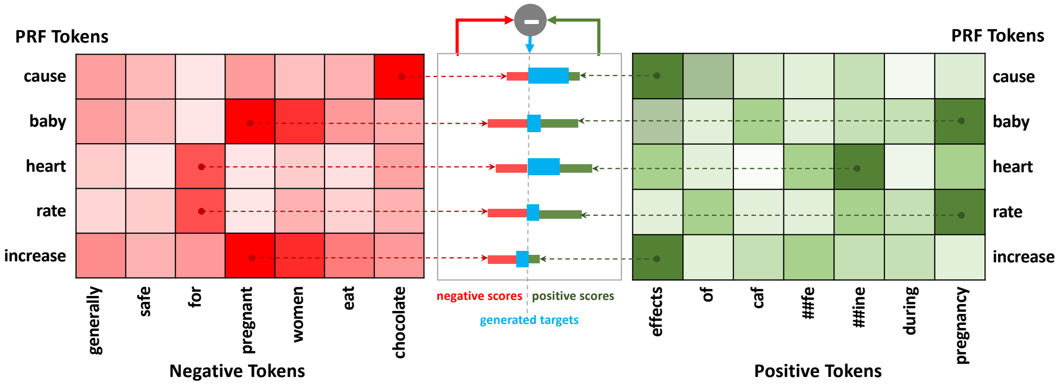


Figure 7.2: Target generation of CWPRF for the training query: "is a little caffeine ok during pregnancy". The target for a PRF token (blue bar) is generated by subtracting (a) the maximum negative similarity score of the PRF token interacting with the tokens from the negative passage (left-hand interaction plot) from (b) the maximum positive similarity of the PRF token interacting with the tokens from the positive passage (right-hand interaction plot).

level representation dense retrieval paradigm exhibit limited retrieval effectiveness Tonellotto and Macdonald (2021a), and hence are not useful for conducting effective pseudo-relevance feedback.

7.1.2 CWPRF Feedback Embedding Weighting

Building on ColBERT, and taking an initially retrieved set of pseudo-relevant feedback passages as input, the CWPRF model aims to predict the importance of each (token-level) feedback embedding in the feedback passages. This is achieved using a separate BERT model instance, which takes a list of input tokens and returns a scalar weight for each token: $\text{CWPRF}(t_1...t_n) = \text{Linear}(\text{BERT}(t_1,..t_n),1)) \in \mathbb{R}^n$.

More specifically, given a document p in the pseudo-relevant set, which is tokenised into a sequence of PRF tokens $p_1, p_2, ..., p_{|p|}$, we employ the ColBERT encoder to obtain its embeddings:

$$\phi_p = \text{ColBERT}(\texttt{[CLS]}, \texttt{[D]}, p_1, ..., p_{|p|}) \in \mathbb{R}^{|p| \times m}. \tag{7.1}$$

Then we obtain the feedback weight for each PRF token using CWPRF which takes the query representations as well as the PRF representations as input:

$$ws = \text{CWPRF}(\overbrace{\texttt{[CLS]}, \texttt{[Q]}, q_1, q_{|q|}}^{\text{query tokens}} \overbrace{\texttt{[D]}, p_1, ..., p_{|p|}}^{\text{PRF tokens}}). \tag{7.2}$$

According to the returned importance score for each of the feedback embeddings in ϕ_p, we identify the highly important ranked embeddings as our expansion embeddings. The expansion embeddings are appended to the original query embeddings to refine the query representation. Note that the original query is included in the invocation of CWPRF$(\cdot)$ – this is by design, to ensure that the CWPRF model considers the relation of the PRF tokens to the original query. However, we ignore the predicted weights of the original query; following ColBERT-PRF, the weights of the original embeddings are assumed to be unchanged. Furthermore, we apply a ReLU upon ws, to ensure that the obtained feedback weights are non-negative. Finally, the score for a document can be calculated as the summation of the weighted MaxSims using the refined query representation:

$$s'(q, f_e, d) = s(q,d) + \beta \sum_{i=1}^{|f_e|} ws_i \cdot \text{MaxSim}(f_{e_i}, \phi_d). \tag{7.3}$$

In particular, the MaxSim function is the Maximum Similarity function used by the ColBERT model and calculated using Equation (2.21).

7.1.3 Training CWPRF

To train CWPRF($\cdot$), we construct a contrastive target for each feedback token. In particular, we use a conventional training file containing triples of $\langle q, d^+, d^- \rangle$, and supplement it with PRF passages, i.e. the passages highly ranked for the original query q, which we assume to be relevant. The aim of our training objective, therefore, is to identify *which* tokens of a feedback passage p result in the positive passage being scored much higher than the negative passage, when the feedback passage is itself treated as the query. Therefore, for each feedback token, and given the positive and negative documents, CWPRF is trained to assign high weights to the tokens that are semantically closer to the tokens occurring in the positive document than those in the negative document. Hence, the target for the i-th PRF token, p_i, is calculated as:

$$t(p_i) = \text{MaxSim}(p_i, d^+) - \text{MaxSim}(p_i, d^-), \tag{7.4}$$

where MaxSim(.,.) measures the semantic similarity between representations, as per Equation (7.3).

The target generation process for CWPRF is illustrated in Figure 7.2. This figure presents the interaction matrices between a PRF document (“cause baby heart rate increase”) obtained from the returned documents list in response to the query: “is a little caffeine ok during pregnancy” compared to the positive and negative document. The shading is indicative of the magnitude of dot product similarity between a PRF embedding and a document embedding, while the highest document embedding for each PRF embedding is indicated with a •. For each PRF embedding, we subtract the negative similarity from the positive similarity, resulting in an importance score for each PRF embedding. In this example, ‘cause’ and ‘heart’ are the most important tokens. These differences are used as targets for learning the CWPRF model.

In addition, we explore two training modes to train CWPRF: All-At-A-Time (AAAT) and One-At-A-Time (OAAT). Suppose, for each training query, the top k ranked documents form the pseudo-relevance feedback set. In the AAAT training mode, we append all the k PRF passages into one single PRF sequence, separated by the `[SEP]` special token, i.e.

$$p_{\texttt{AAAT}} = p^1_1, p^1_2, .., p^1_{|p^1|}, \texttt{[SEP]}, ..., p^k_1, p^k_2, .., p^k_{|p^k|}, \texttt{[SEP]}. \tag{7.5}$$

However, in common with all BERT models, $|p_{\texttt{AAAT}}|$ is limited to 512 tokens, so some tokens may be cut off for large feedback sets. Hence, in the OAAT training mode, each PRF document is regarded as an individual PRF sequence.

$$p^k_{\texttt{OAAT}} = p^k_1, p^k_2, .., p^k_{|p^k|}, \texttt{[SEP]}, \tag{7.6}$$

where k denotes the k-th pseudo-relevance feedback document. The CWPRF training is then conducted for each feedback passage individually.

In-Batch Negative Sampling: In-Batch Negative (IBN) sampling is a technique that has been widely used for training effective dense retrieval models such as DPR (cf. Section 2.3.1). However, it has not previously been applied for query expansion weighting. To promote the discriminative expansion embeddings and suppress the unimportant ones during our target generation, we adapt the idea of in-batch negative (IBN) sampling during the training of CWPRF. Thus, each training sample is equipped with one positive sample and $B-1$ negative samples, where B is the batch size used during training. As a consequence, the target for the i-th PRF token is obtained as:

$$t(p_i) = \mathrm{MaxSim}(p_i, d^+) - \max_{j=1}^{|B-1|} \mathrm{MaxSim}(p_i, d^-_j). \tag{7.7}$$

This ensures that the importance of each feedback embedding for ranking a positive passage is discounted by its presence in all negative passages of the batch. While IBN is commonly used for training ranking models on entire passages, our adaptation focuses instead on the token-level embedding importance.

Loss Functions: CWPRF is trained to assign weights from the target signal using the following objectives. For AAAT training, the loss is computed as follows:

$$\mathcal{L}_{\mathcal{AAAT}} = \frac{1}{N}\sum_{i=1}^{N}(t_{p_i} - ws_{p_i})^2, \tag{7.8}$$

where N is the total number of tokens in the PRF sequence. For the OAAT training mode, we compute the loss for each PRF sequence and add them to obtain the total loss:

$$\mathcal{L}_{\mathcal{OAAT}} = \sum_{j=1}^{k}\left(\frac{1}{N}\sum_{i=1}^{N}(t_{p^j_i} - ws_{p^j_i})^2\right). \tag{7.9}$$

At the inference time, we apply CWPRF consistently with its training mode, i.e. AAAT or OAAT.

7.1.4 Discussion

Connection to ColBERT-PRF: Similar to ColBERT-PRF, CWPRF is implemented in the multiple representation late interaction dense retrieval paradigm. However, in contrast to ColBERT-PRF, CWPRF is a supervised approach, which is tailored for semantic search by selecting and learning the contrastive weights for the discriminate expansion embeddings. The Kendall's τ correlation between the contrastive weights learned by CWPRF and the IDF weights assigned by ColBERT-PRF is only 0.1, which indicates that CWPRF prioritises differently the feedback embeddings. Moreover, compared to ColBERT-PRF, CWPRF has advantages over ColBERT-PRF in that it can identify expansion embeddings that may have low IDF values. It can also avoid the expensive clustering and nearest neighbour lookups used by ColBERT-PRF.

Connection to Learned Sparse Models: In practice, the CWPRF model structure is similar to unexpanded learned sparse retrieval approaches (Dai and Callan, 2020a, Lin and Ma, 2021, Mallia et al., 2021). Importantly, however, the learning objectives are different; learned sparse retrieval optimises for relevance directly, while CWPRF is optimised to identify and weight the most helpful query expansion embeddings.

7.2 Research Questions

In this section, we conduct experiments to address the following research questions:

RQ 7.1: How does the performance of CWPRF, in terms of the retrieval effectiveness, compare to various baselines?

RQ 7.2: How does the performance of CWPRF, in terms of the retrieval efficiency, compare to various baselines?

RQ 7.3: What is the impact of the training techniques on the retrieval effectiveness of CWPRF, namely the in-batch negative sampling, the model initialisation as well as the initial retrieval stage?

RQ 7.4: What is the impact of the parameters of CWPRF, namely the number of feedback passages f_b, the number of the expansion embeddings f_e and the parameter β?

RQ 7.5: How does CWPRF perform across various query types compared with ColBERT-PRF (introduced in Chapter 4)?

7.3 Experimental Setup

In this section, we detail the dataset used in this chapter in Section 7.3.1, then we describe the experimental settings and the baselines in Section 7.3.2 and Section 7.3.3, respectively.

7.3.1 Datasets

We conduct our experiments using the MSMARCO passage ranking dataset (cf. Section 2.5.1). We employ the TREC Deep Learning track 2019 query set as our validation set and use the TREC 2020 query set as our test set due to their dense judgements, which can provide more reliable evaluations of PRF techniques. In addition, we also report the out-of-domain performance of CWPRF on four BEIR datasets (cf. Section 2.5.1).

We evaluate our method using the official metrics of TREC that we introduced in Section 2.5.3, namely nDCG@10, MAP@1000 and Recall@1000. We follow the standard practice of TREC (non-relevant = 0 or 1 and relevant = 2 or 3) for the binary-relevance-based metrics (MAP and Recall). To investigate the extent that the semantic matching, rather than exact token matches occurs when retrieving documents, we also report the semantic match proportion (SMP) for the ColBERT-based system. The calculation of SMP is detailed in Section 4.3.3.4, in Equation (4.5). For significance testing, we use the paired t-test ($p < 0.05$) and apply the Holm-Bonferroni multiple testing correction.

7.3.2 Experimental Implementation

Both the AAAT and OAAT training modes are trained using the MSMARCO triples training set, i.e. the triplets of $\langle q, d^{+} d^{-} \rangle$. Following the settings of ColBERT that we used in Chapter 4 and Chapter 5, we use a ColBERT checkpoint trained using the MSMARCO passage ranking training triplets for 44k steps. We employ the query encoder from the trained ColBERT model to encode the query (the maximum query length is set to 32) and the document encoder to encode the pseudo-relevance feedback documents (the maximum document length is set to 512 for the AAAT training mode and 180 for the OAAT training mode). We set the maximum length to 180 when encoding the positive and negative passages. Following the notations introduced in Section 2.1.2, we use » to denote a retrieval pipeline, for instance `BM25 » ColBERT` indicates applying the ColBERT reranker on the results obtained from BM25. For setting the hyper-parameters of CWPRF, we use the TREC 2019 queries as our validation set; the resulting settings of $f_b = 3$, $f_e = 10$ and $\beta = 5$ are obtained, as reported later in Section 7.4.4. However, we note that $f_b = 3$, $f_e = 10$ is also the recommended setting for ColBERT-PRF (cf. Chapter 4). The high β value indicates the high contribution of the CWPRF identified expansion embeddings for semantic ranking. We further provide the ablations of performing only the expansion embeddings in Section 7.4.4. For both CWPRF and ColBERT-PRF, we perform 5 sets of experiments with varied random seeds for each variant and report the median results.

7.3.3 Baselines

To test the effect of CWPRF, we compare the retrieval effectiveness of a CWPRF-based retrieval system with the following 4 families of retrieval approaches:

- **Sparse Retrieval Systems** (denoted as Sparse in Table 7.1): We compare with the traditional lexical retrieval models, namely BM25 and BM25+RM3 (cf. Section 2.1.1), and both with and without the ColBERT reranker, namely BM25 » ColBERT and BM25+RM3 » ColBERT models;

- **Dense Retrieval Systems (denoted as Dense)**: We compare with both single-representation and multiple-representation dense retrieval models, namely ANCE and ColBERT (cf. Section 2.3);

- **Learned Sparse Retrieval Systems** (denoted as L-Sparse): We compare with SPLADE-v2, DeepImpact and DocT5Query (cf. Section 2.2.3), which are reranked using ColBERT;

- **Dense PRF models** (denoted as D-PRF): we compare with the ANCE-PRF, Vector-PRF and ColBERT-PRF models. We compare our proposed CWPRF model with the more effective ColBERT-PRF Ranker model using the default KMeans clustering (cf. Section 4.2), rather than comparing with the Reranker.

Moreover, when measuring the efficiency of CWPRF, we also compare with the recently proposed variants of ColBERT-PRF, which avoid costly ANN lookups when calculating IDF values for embeddings: KMedoids and KMeans-Closest (cf. Section 4.6).

7.4 Results and Discussion

This section studies the effectiveness as well as the efficiency performance of CWPRF in Section 7.4.1 and Section 7.4.2, respectively. The effects of the various training strategies are investigated in Section 7.4.3. In addition, we examine the impact of the hyperparameters of CWPRF in Section 7.4.4. Next, we investigate the performance of CWPRF according to various query types in Section 7.4.5. Finally, we also provide qualitative analysis of CWPRF in Section 7.4.6.

7.4.1 RQ7.1 - Retrieval Effectiveness of CWPRF

Now we discuss the retrieval effectiveness, in terms of both the in-domain and out-of-domain effectiveness, as well as the matching behaviour of our proposed models compared to various families of baselines.

In-domain Effectiveness: To evaluate the effectiveness of implementing the CWPRF model in a dense pseudo-relevance feedback framework, we compare CWPRF with various families of baselines in Table 7.1. More specifically, the baseline families compared are sparse retrieval models, dense retrieval approaches, learned sparse models as well as the existing dense PRF models, each categorised as a block in Table 7.1.

Table 7.1: Main results on both TREC 2019 and TREC 2020 queries. The superscripts ‘a-j’ denote significant improvements over the indicated baseline model. The highest effectiveness value for each metric is boldfaced. Results not available for significance testing are denoted with ‘-’. † denotes results over-fitted to the test set.

	Systems	TREC 2019 (Validation)				TREC 2020 (Test)			
		MAP	nDCG@10	Recall	Mean-SMP	MAP	nDCG@10	Recall	Mean-SMP
Sparse	(a) BM25	0.2864	0.4795	0.7553	-	0.2930	0.4936	0.8103	-
	(b) BM25 » ColBERT	0.4597	0.6969	0.7553	0.3244	0.4721	0.6891	0.8072	0.3546
	(c) BM25+RM3	0.3108	0.5156	0.7756	-	0.3203	0.5043	0.8423	-
	(d) BM25+RM3 » ColBERT	0.4732	0.7059	0.7756	0.3404	0.4801	0.6866	0.8423	0.3560
Dense	(e) ANCE	0.3715	0.6537	0.7571	-	0.4070	0.6447	0.7737	-
	(f) ColBERT E2E	0.4310	0.6934	0.7892	0.3332	0.4648	0.6871	0.8245	0.3684
L-Sparse	(g) SPLADE-v2 » ColBERT	0.4579	0.6957	0.8723	0.3327	0.4730	0.6794	**0.8987**	0.3682
	(-) DeepImpact » ColBERT	-	0.7220	-	-	-	0.6910	-	-
	(h) DocT5Query » ColBERT	0.5009	0.7136	0.8263	0.3400	0.4733	0.6934	0.8456	0.3618
D-PRF	(i) ANCE-PRF	0.4253	0.6807	0.7912	-	0.4452	0.6948	0.8148	-
	(j) ColBERT-PRF	0.5244	0.7276	**0.8760**	0.3592	0.4904	0.6958	0.8858	0.3837
	(-) Vector-PRF	0.4151	0.6629	0.6962	-	0.4341†	0.6598†	0.7948†	-
Ours	CWPRF-AAAT	**0.5319**^{acefgi}	**0.7444**^{acefgi}	0.8596^{abefi}	0.2814	**0.5136**^{abcefgi}	**0.7246**^{abcdefgj}	0.8783^{abefi}	0.3240
	CWPRF-OAAT	0.5252^{acefgi}	0.7244^{ace}	0.8722^{abefi}	0.2923	0.5049^{abcefgi}	0.7204^{acdefg}	0.8783^{abefi}	0.3265

Among the variants of CWPRF, we observe that when comparing the CWPRF-AAAT and CWPRF-OAAT models (the bottom block), CWPRF-AAAT, which is trained with all PRF passages processed as a single sequence, consistently obtains a higher performance than CWPRF-OAAT, where the PRF sequences are considered individually. This suggests that AAAT provides more relevant context than OAAT for the CWPRF model.

Next, we compare our CWPRF model with other baseline models. Firstly, we observe that the CWPRF models significantly outperform the sparse retrieval models and exhibit marked improvements over sparse-retrieval reranked with the ColBERT reranker. When compared with dense retrieval models, the CWPRF models significantly outperform both types of dense retrieval models. In particular CWPRF exhibits 7.4% (TREC 2019 queries) and 5.5% (TREC 2020 queries) improvements in terms of nDCG@10 compared to the ColBERT E2E model where no expansion embeddings are appended to the original query. This indicates the usefulness of CWPRF for selecting expansion embeddings to augment the query representation. We also compare the CWPRF models with the learned sparse systems, where the document tokens are enriched and reweighted, then applied with a more advanced reranker. We find that both of the CWPRF-AAAT and CWPRF-OAAT models significantly outperform the learned sparse models, indicating the effectiveness of learning the feedback weights and refining the query representation compared with document enrichment.

Finally, when comparing with existing dense PRF models, namely the ANCE-PRF, Vector-PRF and ColBERT-PRF models, we find that the CWPRF models exhibit significant improvements over ANCE-PRF on both query sets and significantly improves over ColBERT-PRF on the TREC 2020 query set. This indicates that our proposed CWPRF approach can select more appropriate expansion embeddings that can help to retrieve more relevant documents, and minimise topic drift.

Out-of-domain Performance of CWPRF on BEIR: Moreover, we examine the performance of

Table 7.2: Effectiveness of CWPRF on BEIR. All scores denote nDCG@10. The best score on a given dataset is boldfaced. † denotes significant differences between CWPRF and the indicated model using paired t-test with $p < 0.05$.

Models	DBPedia	NFCorpus	TREC-COVID	Touché-2020
ANCE	0.265†	0.236†	0.392†	0.291†
ANCE-PRF	0.268†	0.239†	0.430	0.292†
ColBERT	**0.392**	0.316†	0.533	0.307†
ColBERT-PRF	0.387	**0.321**	**0.548**	**0.348**
CWPRF-AAAT	0.385	**0.321**	0.524	**0.348**

the ColBERT and CWPRF (both trained on MSMARCO) in a zero-shot setting, using the BEIR datasets. We choose four datasets from BEIR that have dense judgements (Amati et al., 2004). Table 7.2 reports the performance of CWPRF as well as that of existing dense PRF models on four BEIR benchmarks that we detailed in Section 2.5.1. From Table 7.2, we find that CWPRF shows comparable performance with ColBERT-PRF but with much lower query latency. In addition, CWPRF outperforms ANCE-PRF by a large margin, indicating the superiority of our contrastive weighting method in such zero-shot settings.

Semantic Match Proportion: To further explain the effect of implementing CWPRF for dense query expansion, following the previous chapters in this work, we also report the mean *semantic match proportion* (SMP) values for the models using the ColBERT dense retrieval paradigm in Table 7.1. In particular, SMP quantifies the extent to which a query token exhibits an exact match (matching with the same document token) and a semantic match (matching with different document tokens) in the top-ranked documents. On analysing Table 7.1, we find that, for both query sets, the CWPRF models show lower Mean-SMP values than ColBERT-PRF, implying a more 'focused' retrieval. This is because CWPRF's expansion embeddings correspond to the actual tokens while ColBERT-PRF's expansion embeddings can be the centroid embeddings from clustering. By using more focused embeddings, nDCG@10 is improved compared to ColBERT-PRF. Finally, we present the interaction matrix of CWPRF using the same example query: `why did the us voluntarilay enter ww1?` and passage text that has been used to illustrate the matching behaviour of ColBERT-PRF (cf. Figure 4.6). From Figure 4.6, firstly we observe that the expansion tokens identified by CWPRF in response to the example query are different from the ones identified by ColBERT-PRF (cf. Figure 4.6), where the expansion tokens are `inning`, `attacked`, `##'`, `harbor`, `revolution`, `entered`, `did`, `w`, `m`, `us`. In particular, CWPRF identifies `##wi` as an expansion token for `##w1` while ColBERT-PRF falls short in recognising this good expansion token. Secondly, CWPRF's expansion tokens tend to be more likely to perform the exact matching behaviour than the expansion embeddings identified by ColBERT-PRF. Overall, in response to RQ7.1, these results show that the retrieval effectiveness, for both in-domain and out-of-domain performance, can be markedly improved with the CWPRF feedback weighting technique. Training CWPRF with all PRF passages as one context gives

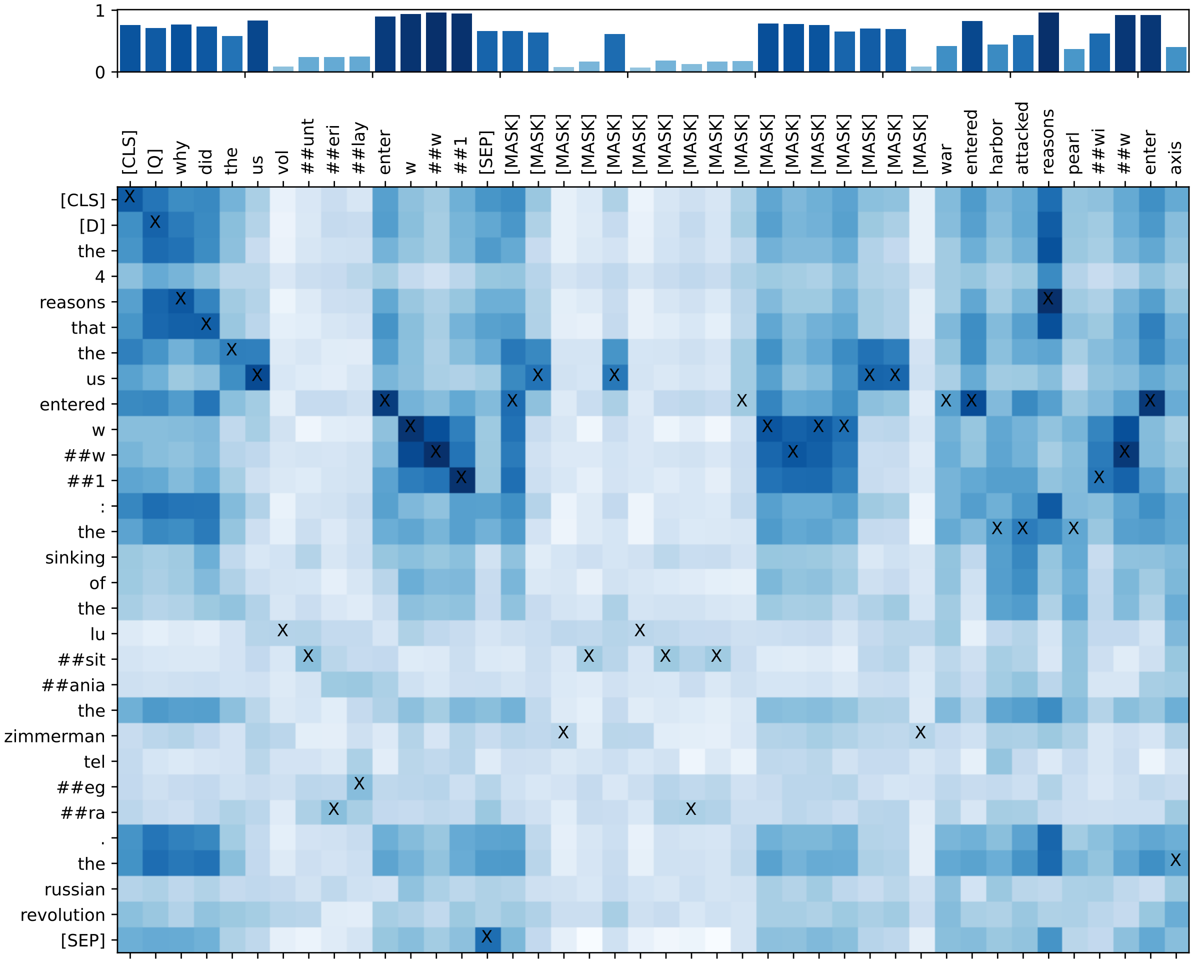


Figure 7.3: CWPRF interaction matrix between query (qid: 106375) and passage (docid: 4337532) embeddings. Notations per Figure 4.6.

more precise retrieval at top ranks. In particular, from Table 7.1, we observe that the CWPRF models achieve the highest nDCG@10 and MAP performances on both query sets and exhibit upto 4.7% improvements on MAP and a 4.1% improvement on nDCG@10 for the TREC 2020 queries compared to ColBERT-PRF.

7.4.2 RQ7.2 - Retrieval Efficiency of CWPRF

Following the three stages described in Figure 7.1, we also report the mean execution time of each stage for various dense PRF systems, including Vector-PRF, ANCE-PRF, variants of ColBERT-PRF with differing efficiency and our CWPRF methods. As Table 7.3 shows, our CWPRF method performs as efficiently as the most efficient ColBERT-PRF variant from Chapter 4 (KMedoids variant) and brings upto 3.06x speedup than the default ColBERT-PRF method (KMeans variant). Although CWPRF needs a longer execution time than Vector-PRF and ANCE-PRF, according to the effectiveness and efficiency tradeoff in Figure 7.4, CWPRF can significantly outperform them without adding much computational cost.

Table 7.3: Mean execution time of dense pseudo-relevance feedback systems. C-PRF represents ColBERT-PRF. Effectiveness and PRF Stage efficiencies are also presented in Figure 1.

Systems	**Mean Execution Time (ms)**			
	Stage 1	PRF Stage	Stage 3	ALL
Vector-PRF	67	4	61	132
ANCE-PRF		111	63	241
C-PRF (KMeans) (default)	387	2997	719	4103
C-PRF (KMeans-Closest)		908	757	2052
C-PRF (KMedoids)		218	744	1349
CWPRF-AAAT		320	710	1417
CWPRF-OAAT		642	714	1743

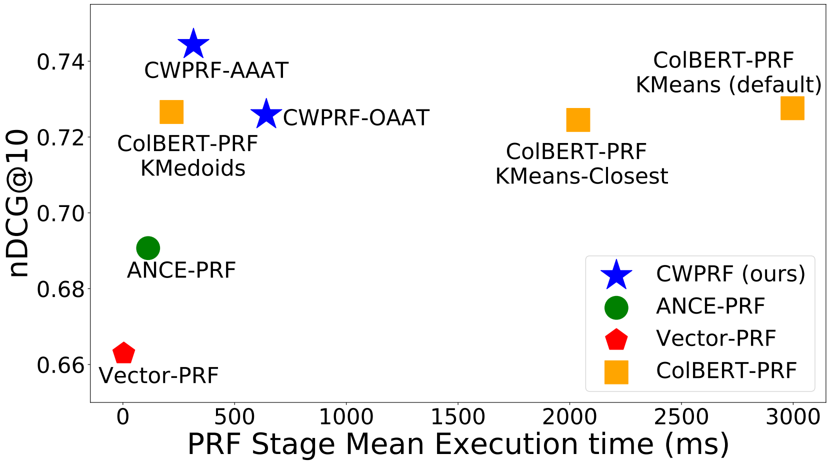


Figure 7.4: Effectiveness (nDCG@10) versus dense PRF stage mean execution time on the TREC 2019 query set.

Figure 7.4 presents the trade-off between the retrieval effectiveness and the mean PRF stage execution time for a variety of existing dense PRF techniques on the TREC 2019 Deep Learning track queries, including Vector-PRF, ANCE-PRF, ColBERT-PRF variants and our proposed CWPRF method. As the figure shows, the default ColBERT-PRF implementation outperforms ANCE-PRF and Vector-PRF in terms of retrieval effectiveness but requires a longer execution time. Meanwhile, our proposed CWPRF achieves the highest nDCG@10 score without requiring high computational cost.

In summary, in response to RQ7.2, our CWPRF model achieves the highest nDCG@10 on the test set among all the compared baselines, while reducing the computational overhead costs compared with previous ColBERT-PRF approaches.

7.4.3 RQ7.3 - Effectiveness of Training Strategies of CWPRF

Next, we inspect the effect of each of the training techniques, namely in-batch negative training, initialisation of the model, different learning objectives and training with PRF passages obtained from different retrieval approaches. Experiments for each training strategy are grouped in Table 7.4.

Effect of In-Batch Negative Sampling: In Table 7.4, we see that training CWPRF with further in-batch negative samples achieves higher retrieval effectiveness on both the TREC 2019 and TREC 2020 query sets, for both the AAAT and OAAT training modes. In practice, more negative training samples for the pseudo-relevance feedback tokens give more opportunity for the model to learn to properly weight unimportant terms in the feedback. For instance, the stopword "it" might occur in the feedback and positive passages, and not in the negative passage, resulting in a high weight. By applying IBNs, there is more chance for "it" to occur in any of the negative passages, reducing its learned target weight, and resulting in a more effective CWPRF model.

Effect of Model Initialisation: Here, we investigate the training from scratch and training with the parameters initialised from an existing learned sparse model, namely uniCOIL (Lin and Ma, 2021). In the second group of Table 7.4, we find that this initialisation for CWPRF can lead to higher performance compared with training from scratch. This is because the pretrained uniCOIL model provides a better starting point than training from scratch. Hence, further fine-tuning with the feedback weighting task dataset results in higher performance.

Effect of Initial Retrieval: Now, we further investigate the training of CWPRF using the PRF passages obtained by sparse retrieval, using BM25, as well as by dense retrieval, using the ColBERT E2E retrieval model. From the final experiment group in Table 7.4, we observe that there is no obvious effectiveness difference between training CWPRF using different initial retrieval systems. Thus, considering the training efficiency, our default CWPRF is trained using the PRF passages obtained from a sparse BM25 initial retrieval.

To summarise, in response to RQ7.3, we find that the in-batch negative training technique and a well-performed initialisation can lead to higher performance of CWPRF, while no significant benefit is observed for different initial training retrieval model.

7.4.4 RQ7.4 - Impact of CWPRF Parameters.

The hyper-parameters for CWPRF are: the number of expansion embeddings f_e and β, which controls the overall contribution of the expansion embeddings. In addition, f_b defines the number of feedback documents used during training and retrieval of CWPRF.

We first vary the f_e and β hyper-parameters during retrieval. Figure 7.5a and Figure 7.5b presents the effectiveness of applying the CWPRF models while varying f_e and β, respectively. Note that $f_e = 0$ or $\beta = 0$ represents the vanilla ColBERT model without any expansion embeddings appended. From Figure 7.5a, we find that for both CWPRF-AAAT and CWPRF-OAAT models, 10 expansion terms give the highest MAP performance. Thus, we set $f_e = 10$ as the default. This echoes the default expansion setting identified for ColBERT-PRF (cf. Section 4.3.3.3). For the β parameter (Figure 7.5b), we find that for both CWPRF-AAAT and OAAT models, MAP performance shows a rising trend as higher $\beta \to 5$ and becomes stable for $\beta > 5$. Indeed

Table 7.4: Performance of CWPRF with different training strategies on the TREC 2019 & 2020 queries. '†' denotes significant improvements over the ColBERT model. The highest value for each metric within each group is boldfaced.

Models	**TREC 2019 (Validation)**		**TREC 2020 (Test)**	
	MAP	nDCG@10	MAP	nDCG@10
ColBERT E2E	0.4310	0.6934	0.4648	0.6871
Effect of In-Batch Negative Sampling (IBN)				
CWPRF-AAAT	0.5168†	0.7331	0.4938	0.7079
CWPRF-AAAT-IBN	**0.5244**†	**0.7332**	0.4966†	0.7045
CWPRF-OAAT	0.5050	0.7064	0.5084†	**0.7125**
CWPRF-OAAT-IBN	0.5151†	0.7269	**0.5094**†	0.7118
Effect of Model Initialisation (Init)				
CWPRF-AAAT-Init	0.5304†	0.7301	0.5125†	0.7184†
CWPRF-AAAT-IBN-Init	**0.5319**†	**0.7444**†	**0.5136**†	**0.7246**†
CWPRF-OAAT-Init	0.5151†	0.7269	0.4948†	0.7112
CWPRF-OAAT-IBN-Init	0.5252†	0.7244	0.5049†	0.7204†
Effect of Initial Retrieval Stage				
CWPRF-AAAT (BM25)	**0.5168**†	0.7331	**0.4938**	0.7079
CWPRF-AAAT (ColBERT)	0.5109†	**0.7346**†	0.4869	0.7002
CWPRF-OAAT (BM25)	0.5050	0.7064	0.5084†	**0.7125**
CWPRF-OAAT (ColBERT)	0.5138†	0.7170	0.4983	0.6904

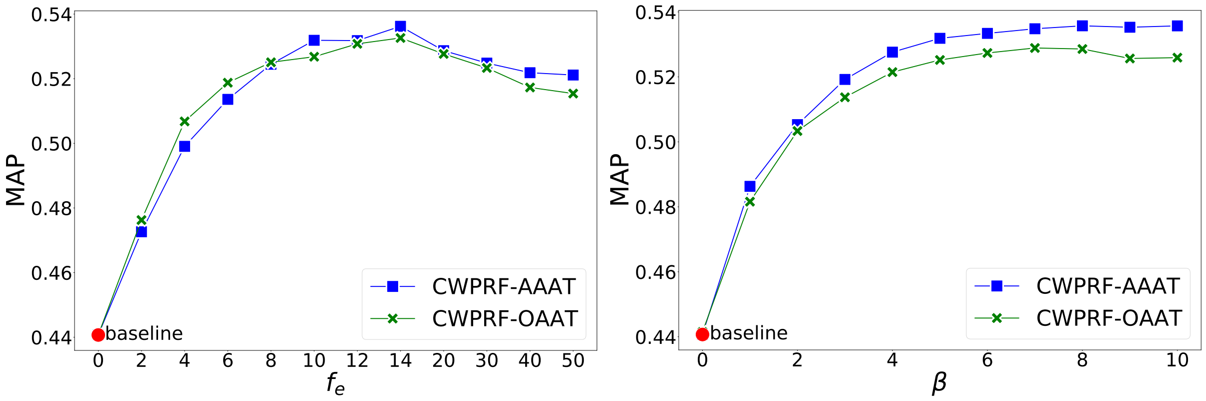
Table 7.5: Contribution of the expansion embeddings of CWPRF on the TREC 2020 test query set. † denotes significant differences over ColBERT using paired t-test with $p < 0.05$.

Systems	MAP	nDCG@10	Recall
ColBERT (only Q)	0.4648	0.6871	0.8245
CWPRF-AAAT (only exp)	0.4824	0.6925	0.8697†
CWPRF-AAAT (Q & exp)	**0.5136**†	**0.7246**†	**0.8783**†
CWPRF-OAAT (only exp)	0.4639	0.6750	0.8600
CWPRF-OAAT (Q & exp)	0.5049†	0.7204†	0.8783†

for $\beta > 5$, it appears that the feedback embeddings are dominating over the original query embeddings. This indicates the high contribution of the selected expansion embeddings during retrieval. Based on this, we set $\beta = 5$ as the default setting of CWPRF.

Indeed, we further quantify the contribution of the expansion embeddings of CWPRF technique and the original query embeddings in respectively in Table 7.5. We find that for CWPRF-AAAT, using only the 10 selected expansion embeddings for reranking, markedly outperforms using the query embeddings alone, which verifies the high contribution of CWPRF selected expansion embeddings.

Furthermore, we study how many PRF passages are needed for CWPRF. We conduct experiments to train both the CWPRF-AAAT and CWPRF-OAAT models with a different number of PRF passages. We note that similar to the setting of the ANCE-PRF model, due to the input length of BERT-based encoders, for the CWPRF-AAAT training, the maximum number of PRF passages



(a) Impact of the number of expansion terms f_e

(b) Impact of the number of expansion terms β

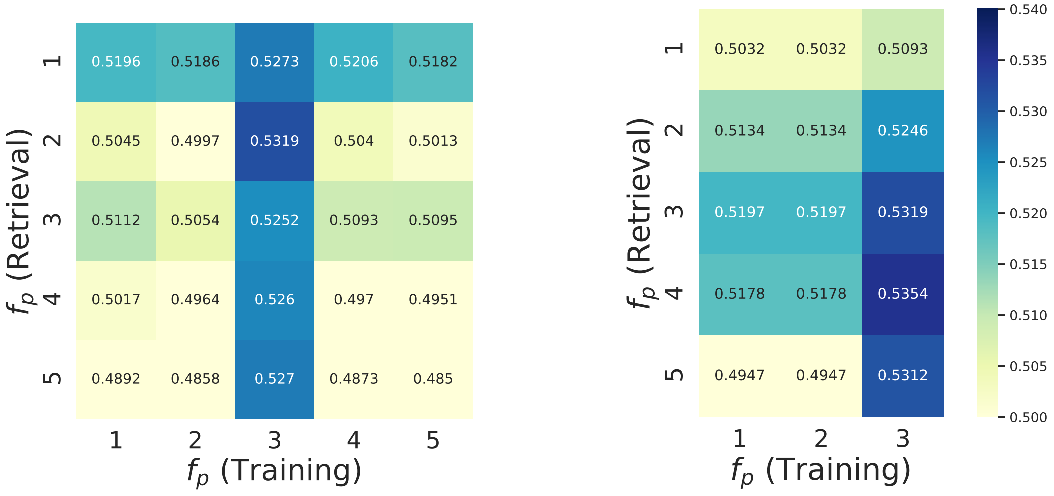
Figure 7.5: Impact of the hyperparameters of CWPRF on the TREC 2019 query set. 'baseline' represents the model without any expansion, i.e. ColBERT E2E.

is set to 3. On the other hand, for the OAAT training mode, as each PRF document is treated independently, there is no such requirement. The nDCG@10 results are presented in Figure 7.6. We observe that for CWPRF-OAAT, three feedback documents employed for training alone or evaluation alone give higher performance than other f_b values. Overall, the combination of $f_b = 3$ for both training and retrieval gives the highest performance. In addition, for CWPRF-AAAT, we find that a high MAP performance is achieved by training with only the top two PRF passages. However, this is not stable, as during retrieval, more PRF passages are needed under this setting. This indicates the model might not be trained enough. Moreover, we observe a similar trend for $f_b = 3$ used for both training and retrieval. Thus, based on this observation, we suggest to set $f_b = 3$ as the default for the training and evaluation of CWPRF.

To summarise, in response to RQ7.4, we recommend using the parameters where the number of feedback passages $f_b = 3$ and the number of expansion embeddings $f_e = 10$ as well as $\beta = 5$ for CWPRF. Note that $f_b = 3$ and $f_e = 10$ are also the default setting for ColBERT-PRF reported in Section 4.3.3.3.

7.4.5 RQ7.5 - Performance of CWPRF on Different Query Types

We further investigate the performance of the CWPRF models compared to ColBERT on different query types using the query taxonomy of Bolotova et al. (2022). Specifically, we combine the TREC 2019 and TREC 2020 queries to create a single query pool, consisting of 97 queries. Then, the merged queries are classified using a trained query category classifier according to the query taxonomy introduced by Bolotova et al. (2022). Figure 7.7a and Figure 7.7b illustrate the absolute difference in performance between the CWPRF-AAAT model and the ColBERT-PRF model in terms of MAP and nDCG@10, respectively. Similarly, Figure 7.7c and Figure 7.7d provide comparisons for the CWPRF-OAAT model against ColBERT-PRF. From Figure 7.7, it is evident that CWPRF-AAAT demonstrates improvement across all query types in terms of MAP and



(a) CWPRF-OAAT.

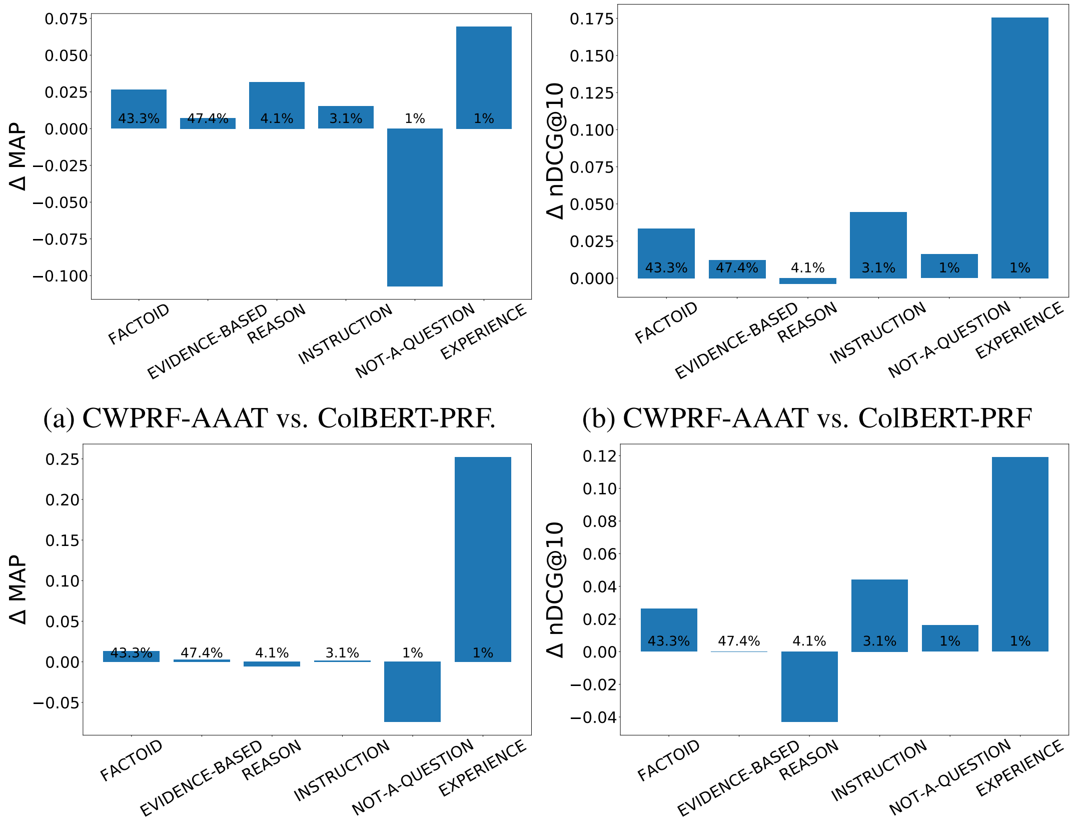
(b) CWPRF-AAAT.

Figure 7.6: Impact of the number of feedback passages f_b during training (x-axis) and retrieval (y-axis) for CWPRF, in terms of MAP on the TREC 2019 query set.

nDCG@10, except for the NOT-A-QUESTION type. However, it is worth noting that the number of queries belonging to the NOT-A-QUESTION type is quite low, comprising only approximately 1% (a single query) of the total. Similarly, we observe that CWPRF-OAAT also enhances performance across various query types, except for the single NOT-A-QUESTION type in terms of MAP, and the REASON type (with a ratio of approximately 4.1%) in terms of nDCG@10. To summarise, in response to RQ7.5, we find that CWPRF exhibits higher retrieval effectiveness across various types of queries. These observations further highlight the effectiveness and robustness of our proposed CWPRF models compared to ColBERT-PRF across diverse query types.

7.4.6 Qualitative Analysis

Table 7.6 presents an example of the expansion tokens identified by CWPRF and the ColBERT-PRF technique as well as their retrieved top-ranked document. We observe that the two comparing methods can generate some expansion tokens in common but not necessarily receive the same weights. In particular, compared to the ColBERT-PRF model, CWPRF can bring a highly relevant document (Label=2) to the top rank, by expanding with tokens: “revision” and “allows”, which are helpful in retrieving the more relevant document (indicated by their darker shading). Indeed, this superior ability to retrieve highly relevant documents at high ranks is more useful in a real-life retrieval scenario. Unexpectedly, “allow” and “allows” are identified by CWPRF as important expansion tokens. This indicates that CWPRF can take the context into account – more so than IDF. The second example in Table 7.6 is selected from a case when CWPRF underperforms ColBERT-PRF. Indeed, while CWPRF experiences a performance drop compared to ColBERT-PRF, it can still retrieve a document with label 3 at the top rank. This indicates the benefits of our contrastive weighting technique for bringing more relevant documents to the top positions.



(a) CWPRF-AAAT vs. ColBERT-PRF.

(b) CWPRF-AAAT vs. ColBERT-PRF

(c) CWPRF-OAAT vs. ColBERT-PRF.

(d) CWPRF-OAAT vs. ColBERT-PRF.

Figure 7.7: Performance of CWPRF compared to ColBERT-PRF across different types of queries according to the query type taxonomy proposed by Bolotova et al. (2022). The percentage of queries within each query type, relative to the total number of queries in the query pool, is indicated within each bar.

Overall, we see that CWPRF can select more useful expansion embeddings to help bring more relevant documents on top, which would be more useful when implementing in a retrieval system in a real-life scenario.

7.5 Conclusions

To summarise, in the proposed thesis statement in Section 1.1, we hypothesised that we can employ the pseudo-relevance feedback information to train an effective dense query expansion model. Therefore, in this chapter, we proposed a deep language model-based contrastive weighting approach (CWPRF) for selecting useful query expansion embeddings and calibrating their expansion weights for semantic search. In particular, CWPRF is trained with a contrastive objective to learn to assign a high weight for feedback embeddings that can distinguish relevant documents from non-relevant documents. During retrieval, the embeddings of tokens appearing in the feedback documents that CWPRF predicts to be important are appended to the query embeddings. Extensive experiments performed on two query sets demonstrate that our proposed

Table 7.6: Example of the expansion tokens identified by the CWPRF and ColBERT-PRF approaches, as well as the top returned passage for each approach after applying PRF. Tokens with a darker red contribute more to nDCG@10.

Approach	CWPRF > ColBERT-PRF	QID 156498: Query: **do google docs auto save**	
CWPRF	Expansion tokens	doc google save ##s allows revision automatically deleted allow just	
	Top returned passage after PRF	DOCNO: 104801 TEXT: **Allow Google Docs** to **automatically save** your **document**. As you add new content to your **Google Doc**, the **changes** you make to the **document** are **automatically saved** to your drive. Next to the“Help” tab at the top of your screen, you will see light gray text.	Label=2
ColBERT-PRF	Expansion tokens	##’ doc automatically google document save saves drive changes back	
	Top returned passage after PRF	DOCNO: 104803 TEXT: Allow **Google Docs** save and sync your changes **automatically**. In the offline application, **Google Drive automatically saves changes** made to a **document** every few seconds. When your computer connects to the internet, the **Google Drive** application will function like its online counterpart.	Label=1
	CWPRF < ColBERT-PRF	QID 67316: Query: **can fever cause miscarriage early pregnancy**	
CWPRF	Expansion tokens	fever cause pregnancy mis ##carriage increases baby temperature causing birth	
	Top returned passage after PRF	DOCNO: 6680964 TEXT: 1 A **temperature above 103F** (39.50C) during **early weeks of pregnancy** (usually the first trimester) may be responsible for a **miscarriage**, spinal cord or mental defects in the baby. **Fever in early pregnancy** may cause more harm than fever in late pregnancy.	Label=3
ColBERT-PRF	Expansion tokens	defects ##’ ##ping bath trim fever studies pregnancy early during	
	Top returned passage after PRF	DOCNO: 7348851 TEXT: A **temperature higher than 100.4 degrees** Fahrenheit – or the illness causing the fever – could harm both you and your developing **baby**. **A high fever** increases the risk of **birth defects or miscarriage in early pregnancy**. The higher the fever and the longer it lasts, the higher the risk. If you want to lower your fever without using medicine like acetaminophen – or just don’t have any on hand – you can try these methods: 1 Lie down and place a cool, damp washcloth on your forehead. 2 Take a lukewarm tub bath or sponge bath.	Label=3

CWPRF approach can significantly outperform the ColBERT dense retrieval model. In particular, CWPRF significantly improves over ColBERT-PRF by 4.1% in terms of nDCG@10 (cf. Table 7.1) on the TREC 2020 query set without requiring a high computational cost.

Overall, the findings for the CWPRF can be summarised as follows: (i) In terms of the retrieval effectiveness of CWPRF, for both in-domain and out-of-domain performance, we found that our CWPRF exhibits markedly improvements over various baselines and achieves the highest nDCG@10 and MAP performances on both query sets; (ii) In terms of the tradeoff between the retrieval effectiveness and efficiency, we found that CWPRF model achieves the highest nDCG@10 score on the test set while reducing the computational overhead costs compared with the various ColBERT-PRF approaches introduced in Chapter 4; (iii) Moreover, CWPRF enhances performance across various query types in terms of nDCG@10.

Based on these findings in this chapter as well as the previous chapter, we can conclude that the pseudo-relevance feedback information can be leveraged in both unsupervised, i.e. our proposed ColBERT-PRF model, and supervised way, i.e. our proposed CWPRF model, for effective dense query expansion. Both our proposed ColBERT-PRF and CWPRF models exhibit significant improvements over existing dense query expansion methods but perform in a different method.

However, there are also some limitations to the CWPRF method. It is unclear how it may be adapted for the single-representation dense retrieval PRF model. In addition, in this work, we did not test the effect of the hard negative sampling and the number of negative samples for CWPRF. Finally, while we have focused on passage retrieval, longer document retrieval can be addressed through splitting documents into passages during indexing, retrieval and PRF, and applying a max-passage aggregation (Dai and Callan, 2019b) to obtain a document ranking.

For future work, we will also consider a hybrid approach to incorporate both the learned weights produced by CWPRF and the statistical information in the expansion embedding identification process. In the next chapter, we will close this thesis by summarising the results and conclusions of each chapter and discussing possible future directions uncovered by this work.

Chapter 8

Conclusions and Future Work

8.1 Contributions and Conclusions

This thesis addressed the challenges of using pseudo-relevance feedback techniques for more effective sparse and dense retrieval. Specifically, we argued that the pseudo-relevance feedback information can be used in neural-based models to improve retrieval effectiveness, for both sparse and dense retrieval. In Chapter 1, we argued that the existing neural pseudo-relevance feedback models have the following challenges:

- **Challenge 1:** How to use the pretrained knowledge of sequence-to-sequence generative language models to generate query reformulations to further enhance adhoc retrieval effectiveness?

- **Challenge 2:** How to effectively implement a pseudo-relevance feedback mechanism for effective dense retrieval?

- **Challenge 3:** Most dense retrieval models are shown to face challenges when it comes to out-of-domain evaluations. However, the question arises: How can we employ high-quality external knowledge on the dense query expansion models, such as ColBERT-PRF, to improve the out-of-domain performance of dense retrieval models?

- **Challenge 4:** While the default ColBERT and ColBERT-PRF models have only been applied to the BERT model and its corresponding WordPiece tokeniser, the extent that ColBERT and ColBERT-PRF generalise across various types of pretrained models and the tokenisation method is still under-investigated.

- **Challenge 5:** The ColBERT-PRF models perform dense query expansion in an unsupervised manner, depending on heuristical techniques such as clustering and IDF statistics. A challenge is therefore how to perform effective dense expansion in a supervised way.

To address the above five challenges, we have proposed various models in this thesis. In the following, we discuss our main contributions and conclusions in addressing these challenges:

- **Conclusion 1: Generative Query Reformulation for Effective Adhoc Search.** To address the first challenge, we cast the query reformulation task as a text generation task. Accordingly, we explored two possible generative query reformulation frameworks, GenQR and GenPRF (cf. Chapter 3). In particular, models under the GenQR framework directly take a query as input, while models under the GenPRF framework also incorporate contextual information extracted from the pseudo-relevant feedback documents. Moreover, under each framework, we investigated both fine-tuning and direct prompting methods to leverage the learned knowledge of T5 and FLAN-T5, respectively (cf. Section 3.1). In particular, for the T5-based query reformulation models, we investigated the use of weakly supervised query pairs to fine-tune the T5-based query reformulation models (cf. Section 3.1.3). The results of Section 3.4.1 to 3.4.3 demonstrate that the reformulated queries generated by the GenQR and GenPRF models can significantly improve over the original query and statistical query expansion approaches as well as show comparable performance to existing neural-PRF models.

- **Conclusion 2: Semantic PRF for Effective Dense Retrieval.** To address challenge 2, we proposed ColBERT-PRF, where we implement the pseudo-relevance feedback mechanism on multiple representation dense retrieval (cf. Chapter 4). More specifically, in Section 4.2, ColBERT-PRF applies clustering to the embeddings occurring in the pseudo-relevant set, and then identifies the most discriminative embeddings among the cluster centroids. The identified expansion embeddings are appended to the original query as the refined query representation. Experiments in Section 4.3 & Section 4.4 shown that pseudo-relevance feedback information from the top-returned passages in multiple representation dense retrieval is beneficial for improving the retrieval effectiveness on passage retrieval and document retrieval, respectively. In addition, we investigated various techniques to measure the informativeness of expansion embeddings of ColBERT-PRF in Section 4.5, namely the statistical methods: IDF and ICTF, and on embedding coherency, namely Mean Cosine Similarity. Finally, we investigated the various efficient variants of ColBERT-PRF in Section 4.6, the experiment results demonstrate that the trade-off of the retrieval effectiveness and efficiency of ColBERT-PRF can be attained using different clustering techniques and/or candidate selection techniques based on approximate scoring.

- **Conclusion 3: Dense External Expansion.** To address the third challenge, in Chapter 5, we studied the effectiveness of external dense expansion on dense retrieval. More specifically, we investigated external expansion when mixing sparse & dense retrieval paradigms (including both single representation and multiple representation dense retrieval) in Section 5.2. We conducted experiments on two classical TREC test collections (Robust04 & WT10G) as well as four BEIR datasets, namely DBPedia, NFCorpus, TREC-COVID and Touché-2020. Moreover, we investigated different frameworks performing external expansion for zero-shot retrieval, namely (a) *dense external expansion for sparse retrieval*, (b) *dense external expansion for dense retrieval* and (c) *sparse-obtained external feedback for dense retrieval*. The results presented in Section 5.5 show that pseudo-relevance feedback, in the form of external expansion, can result in effective zero-shot retrieval.

- **Conclusion 4: From ColBERT-PRF to Col⋆-PRF.** To address challenge 4, in Chapter 6, we studied the effectiveness of multi-representation dense retrieval as well as the dense query expansion models with different pretrained models with different tokenisation techniques (cf. Section 6.1). In particular, experimental results in Section 6.1.3 demonstrate that ColBERT and ColBERT-PRF can generalise upon various pretrained language models as Col⋆ and Col⋆-PRF, respectively. Moreover, we further examined the matching behaviour of the multiple representation dense retrieval models before and after the implementation pseudo-relevance feedback mechanism in Section 6.2. The extensive experimental analysis yields new findings that can shed light on the more effective dense retrieval model design and retrieval, including (i) applying the Col⋆ and Col⋆-PRF models with the BPE tokeniser is more likely to perform semantic matching than the more common ColBERT model; (ii) among various salient token families, all of the (weighted) contextualised late interaction models perform semantic matching, particularly for low IDF tokens and stopwords tokens; (iii) performing only exact matching and the special token matching contribute more than only semantic matching to the overall retrieval effectiveness.

- **Conclusion 5: Learning Feedback Weights for Dense Query Expansion.** Finally, to address the fifth challenge, we propose a contrastive weighting method, called CWPRF, to select and weight the usefulness of the feedback embeddings for dense expansion in Chapter 7. In particular, CWPRF performs in a supervised way by training with the contrastive objective allowing it to learn the effective weights for expansion embeddings that are tailored for the semantic ranking task (cf. Section 7.1). More specifically, for each feedback token, we constructed a contrastive objective, where, given positive and negative documents, CWPRF is trained to assign high weights to the tokens that are semantically closer to tokens occurring in the positive document than to those in the negative document. Experimental results presented in Section 7.4 & Section 7.4.2 demonstrated that CWPRF can achieve significantly higher retrieval effectiveness but with less execution time than the default ColBERT-PRF. In addition, extensive empirical evaluations in Section 7.4.3 & Section 7.4.4 demonstrate how to effectively train our CWPRF in a supervised way. Finally, we showed that CWPRF enhances performance across various query types in terms of nDCG@10 (cf. Section 7.4.5).

Next, based on the results obtained in Chapters 3 to 7, we now validate our thesis statement posed in Section 1.1. Our thesis stated that we can use the neural models based on the pseudo-relevance feedback information to improve the retrieval effectiveness of the sparse as well as the dense retrieval. In the following, we discuss the corresponding experimental results and observations that validate our proposed thesis statement.

- **Claim 1:** *Pseudo-relevance feedback information can be used by a sequence-to-sequence neural model to generate more effective query reformulations for sparse retrieval.* Our experiments in Chapter 3 validated this claim by showing that our proposed Generative Query Reformulation models can significantly outperform the retrieval results based on the original query as well as the various baseline models (cf. Table 3.6). Specifically, Table 3.9 provides the qualitative analysis of our proposed generative query reformulation models that they can produce queries that are more precise than RM3 (in terms of nDCG@10) while also enhancing Recall (cf. Table 3.6).

- **Claim 2:** *Moreover, applying pseudo-relevance feedback on contextualised embeddings can refine the query representation for multiple representation dense retrieval, in particular, for the ColBERT model.* We validated this claim in Chapter 4, where we propose ColBERT-PRF by implementing the pseudo-relevance feedback mechanism on the multiple representation dense retrieval. In particular, ColBERT-PRF employs the clustering technique to find the representative centroid embeddings based on the pseudo-relevance feedback document embeddings. Then ColBERT-PRF uses IDF as an informativeness measurement method to identify the most suitable expansion embeddings among the extracted representative centroid embeddings. Extensive experiments results demonstrated that ColBERT-PRF can significantly outperform the existing various families of baselines for passage retrieval (cf. Table 4.1) and long document retrieval (cf. Table 4.4 for MSMARCO document ranking dataset) & Table 4.5 for Robust04 document ranking dataset). In particular, our ColBERT-PRF outperforms the ColBERT E2E model by 26% and 10% on TREC 2019 and TREC 2020 passage ranking query sets (cf. Table 4.4). Furthermore, efficient variants of ColBERT-PRF using the approximate scoring technique and/or different clustering algorithms can bring upto 4.54x speedup without compromising the retrieval effectiveness (cf. Table 4.6).

- **Claim 3:** *Furthermore, performing external pseudo-relevance feedback to refine the query representation can improve the zero-shot performance for both sparse and dense retrieval.* We argue that we validated this claim in Chapter 5 by proposing the external dense expansion technique and performing the zero-shot evaluation. In particular, we examined the external expansion uses both the sparse and dense retrieval paradigms in different settings, see Section 5.2. More specifically, when experimenting with the setting denote as "Dense External, Sparse Retrieval" in Table 5.3, we observed that external sparse expansion exhibits similar performance to the target expansion sparse retrieval models, where external dense expansion can bring significant improvements over sparse retrieval models with expansion only performed on the target (12% improvement for Robust04 in nDCG@10: 0.432 → 0.482 and 28% for WT10G: 0.324 → 0.420 in Table 5.3). In addition, when experimenting with the setting "Dense External, Dense Retrieval", we find that (i) external expansion using ColBERT can significantly improve over the dense retrieval models as well as the dense retrieval with target query expansion (7% improvement for Robust04 in nDCG@10: 0.447 → 0.477 and 21% 0.328 → 0.397 in Table 5.4); (ii) external expansion using ANCE can improve the retrieval effectiveness of the dense retrieval (by 20% for Robust04 on nDCG@10: 0.324 → 0.388 and by 29% for WT10G: 0.224 → 0.289 in Table 5.7); (iii) performing dense external expansion using ColBERT or ANCE can result in further improvements on four BEIR datasets in Table 5.5. Finally, we investigated the setting "Sparse External, Dense Retrieval", and found that sparse external expansion brings limited useful information for ColBERT to improve the followed-up dense retrieval effectiveness, and that even applying sparse external retrieval as an initial stage can bring useful feedback documents to improve over the dense retrieval on the target collection (cf. Table 5.4 & Table 5.7).

- **Claim 4:** *In addition, our key ColBERT-PRF model can be effectively extended to various forms of late interaction dense retrieval models.* In Chapter 6, we have validated this Claim. In particular, we firstly extended ColBERT to Col⋆ by instantiating the late interaction mechanism with various pretrained models using different types of tokenisation techniques. Then, we generalised our proposed ColBERT-PRF technique to Col⋆-PRF. Furthermore, we extensively evaluated the retrieval effectiveness of the various extended Col⋆-PRF models, and found that (i) the implementation of the ColBERT-PRF pseudo-relevance feedback technique will alter the proportion of semantic matching, for example, the SMP values for ColRoBERTa-PRF and ColALBERT-PRF may decrease compared to those of ColRoBERTa and ColALBERT models, while opposite observations can be made for ColBERT-PRF and ColminiLM-PRF models, where the SMP values may increase (cf. Table 6.4); (ii) inconsistent to Col⋆ models, low IDF tokens tend to perform semantic matching in Col⋆-PRF models, however, the extent that a type of tokens' semantic matching proportion alters depends on the implementation of the pseudo-relevance feedback technique (cf. Table 6.9); (iii) special token matching contributes the most to the overall retrieval effectiveness of the Col⋆-PRF models. Conversely, semantic matching contributes the least to the overall retrieval effectiveness (cf. Table 6.10).

- **Claim 5:** *Finally, pseudo-relevance feedback information can be used to train a deep language model-based feedback weighting model for identifying the discriminating expansion embeddings for query reformulation.* We argue that we have validated this claim in Chapter 7, where we proposed a deep language model-based contrastive weighting approach (CWPRF) for selecting useful query expansion embeddings for effective dense retrieval (cf. Section 7.1). In particular, for each feedback token, we construct a contrastive objective, where, given positive and negative documents, CWPRF is trained to assign high weights to the tokens that are semantically closer to tokens occurring in the positive document than to those in the negative document. Moreover, we conducted extensive experiments to evaluate the proposed CWPRF model and found that in terms of retrieval effectiveness, the CWPRF approaches achieve the highest nDCG@10 and MAP performances on both query sets and exhibit upto 4.7% improvements on MAP and a 4.1% improvement on nDCG@10 for the test queries compared to ColBERT-PRF (cf. Table 7.1). In addition, in terms of efficiency, as shown in Table 7.3, our CWPRF method performs as efficiently as the most efficient ColBERT-PRF variant (KMedoids variant cf. Section 4.6) and brings upto 3.06x speedup than the default ColBERT-PRF method (KMeans variant, cf. Section 4.6). Overall, we find CWPRF model achieves the highest nDCG@10 on the test set among all the compared baselines, while reducing the computational overhead costs compared with previous ColBERT-PRF approaches.

In summary, we have validated each of the proposed claims in our thesis statement in Section 1.1. We have shown that we can use the pseudo-relevance feedback technique in the neural language models to improve sparse retrieval effectiveness using the generative query reformulation models and to improve dense retrieval effectiveness using ColBERT-PRF models. In addition, we have shown that we can employ the external dense expansion method to further improve zero-short retrieval evaluation. Furthermore, we have shown that ColBERT-PRF can be generalised to Col⋆-PRF across various pretrained language models with different tokenisation techniques. Finally, we have shown that we can employ the pseudo-relevance feedback information to learn the feedback weights for effective dense query expansion. In addition to the significant improvements of the proposed pseudo-relevance feedback techniques, observed on various test collections, existing studies have indicated that evaluation metrics such as nDCG are highly correlated with user preferences (Radlinski and Craswell, 2010, Sanderson et al., 2010). Therefore, the proposed methods are likely to benefit user's satisfaction with the search engine. In the following, we describe several further directions for neural pseudo-relevance feedback models.

8.2 Directions for Future Work

In this section, we discuss possible future directions that could further benefit from the neural pseudo-relevance feedback models for effective information retrieval.

PRF-Trained Dense Retrieval Models: On analysing the training objectives employed in the neural retrieval model, for instance, the commonly used negative log-likelihood loss, many works, such as ANCE (Xiong et al., 2021) and ColBERT-v2 (Santhanam et al., 2022) found that the negative sampling strategy is critical for effective training. As we have introduced in Section 2.3.1, different negative sampling strategies have been studied, for instance, random negative sampling, in-batch negative sampling and the dynamic hard negative sampling techniques. However, even with the hard negative samples, it is hard to train an effective dense retrieval model due to the high possibility of a large amount of *false negatives* or unlabeled positives (Qu et al., 2020) occurring in the training dataset. For instance, in the popular used MSMARCO training dataset (introduced in Section 2.5.1), each query in the training dataset has an average of 1.1 judged positive passages, while Qu et al. (2020) showed that upto 70% of the top returned passages are actually positive passages that were not annotated by the assessors. Thus, these false-negative samples can hinder the training of an effective dense retrieval model using the cross-entropy object upon triplets of $\langle query, relevant, non\text{-}relevant \rangle$. Thus, in the future, we would like to explore a more effective training strategy to incorporate the pseudo-relevance feedback documents, which are often regarded as false negatives in the negative samples, as additional training examples for improving the dense retrieval model. As a result, a PRF-trained dense retrieval model is expected to have better query and document encoders to produce more discriminating representations thus achieving a better ranking performance.

Selective Dense-PRF: Although pseudo-relevance feedback techniques have shown their usefulness for both sparse and dense retrieval, most query expansion models expand and re-weight the original query using the same number of expansion terms regardless of the type of the input query. For instance, in RM3 and ColBERT-PRF, the default setting for the number of expansion terms/embeddings is 10. A thread of research focuses on performing query expansion selectively based on the input query. Several selective query expansion models have shown that retrieval effectiveness can be improved by performing query expansion selectively (Amati et al., 2004, Carmel and Yom-Tov, 2010, Cronen-Townsend et al., 2004, He and Ounis, 2004). However, these methods are based on the sparse retrieval paradigm and no works have investigated such benefit would also occur for dense-PRF models. Thus, we would like to investigate the effect of retrieval effectiveness in performing dense-PRF in a selective way. Furthermore, we would like to investigate the effect of the retrieval effectiveness using a different number of expansion embeddings/tokens in response to different types of queries.

Generative Dense Query Expansion: In Chapter 3, we introduce the sequence-to-sequence model-based generative query reformulation and generative pseudo-relevance feedback techniques for sparse retrieval. However, these works treat the user query as a single unit to input into the prompt of pretrained large language models for the query reformulation task. Recent research has indicated that breaking the task into multiple intermediate steps for prompting LLM can yield significant improvements in performance. For instance, the Chain-of-Thought (CoT) method (Wei et al., 2022b) was recently proposed to introduce a chain of intermediate thoughts to bridge the user input and the LLM output to solve a math problem. In addition, the very recent Tree-of-Thought (ToT) (Yao et al., 2023) technique maintains a tree of the intermediate steps for solving a problem. However, the applicability and effectiveness of these advanced prompting methods have not been observed for the information retrieval task yet. Moreover, studies by Bendersky et al. (2010) have shown that modelling query concepts through term dependencies analysis, especially for long complex user queries, can further enhance retrieval effectiveness. Thus, we would like to investigate a Concept-of-Thought method for prompting the LLMs to generate more diverse and well-represented expansion terms for generative query reformulation tasks.

Dense QE for Single Representation: ANCE-PRF and Vector-PRF, which are introduced in Section 2.4.3, have tried to implement the pseudo-relevance feedback mechanism to the single representation dense retrieval. However, the improvement brought by the additional PRF information is limited. It is because, on the one hand, ANCE-PRF performs query reformulation implicitly by training a new query encoder which conditions on the query as well as the PRF documents as the input. On the other hand, Vector-PRF reformulates the query representation heuristically via the weighted combination of the query and PRF document representations. Moreover, a single representation dense retrieval encodes the whole query or document as a single embedding, it is impossible to disentangle the actual expanded information for a query. Motivated by the TCT-ColBERT models (Lin et al., 2020a, 2021b), which also belong to the single representation dense retrieval family but the query and document representations are obtained by applying the average pooling over the query or document token-level representations rather than using the `[CLS]` embeddings. Therefore, we would like to investigate a single dense PRF method that can explicitly refine the query representation. In particular, we would like to first identify the useful tokens from the token-level embeddings, and then employ these selected useful information to refine the query representation. By doing this, we can inspect the importance of the tokens that can be used for query expansion.

8.3 Concluding Remarks

This thesis has addressed a challenging task: the information retrieval task. In particular, this thesis contributed to the development of effective information retrieval models based on pseudo-relevance feedback information. Specifically, we have shown that the pseudo-relevance feedback information can be used by the sequence-to-sequence generative models to refine the query representation for sparse retrieval. Moreover, pseudo-relevance feedback information can also be used to refine the query representation for more effective dense retrieval models in both unsupervised and supervised manners. However, in Section 8.2, we have identified a number of interesting directions in this field. This thesis has laid a solid foundation and provided strong motivation for further exploring these research directions in the future. We believe that pseudo-relevance feedback information will continue to benefit the future development of the effective information retrieval field.

Bibliography

Abdul-Jaleel, N., Allan, J., Croft, W. B., Diaz, F., Larkey, L., Li, X., Smucker, M. D., and Wade, C. (2004). Umass at trec 2004: Novelty and hard. In *Proceedings of TREC*.

Acquavia, A., Macdonald, C., and Tonellotto, N. (2023). Static pruning for multi-representation dense retrieval. In *Proceedings of DocEng*, pages 1–10.

Amati, G. (2003). Probability models for information retrieval based on divergence from randomness ph.d. thesis. *University of Glasgow*.

Amati, G. (2006). Frequentist and bayesian approach to information retrieval. In *Proceedings of ECIR*, pages 13–24.

Amati, G., Ambrosi, E., Bianchi, M., Gaibisso, C., and Gambosi, G. (2007). Fub, IASI-CNR and university of tor vergata at TREC 2007 blog track. In Voorhees, E. M. and Buckland, L. P., editors, *Proceedings of TREC*.

Amati, G., Carpineto, C., and Romano, G. (2004). Query difficulty, robustness, and selective application of query expansion. In *Proceedings of ECIR*, pages 127–137.

Amati, G. and Van Rijsbergen, C. J. (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)*, 20(4):357–389.

Arabzadeh, N., Yan, X., and Clarke, C. L. (2021). Predicting efficiency/effectiveness trade-offs for dense vs. sparse retrieval strategy selection. In *Proceedings of CIKM*, pages 2862–2866.

Arapakis, I., Bai, X., and Cambazoglu, B. B. (2014). Impact of response latency on user behavior in web search. In *Proceedings of SIGIR*, pages 103–112.

Arthur, D. and Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. In *Proceedings of SODA*, page 1027–1035.

Asadi, N. and Lin, J. (2013). Effectiveness/efficiency tradeoffs for candidate generation in multi-stage retrieval architectures. In *Proceedings of SIGIR*, pages 997–1000.

Bai, Y., Li, X., Wang, G., Zhang, C., Shang, L., Xu, J., Wang, Z., Wang, F., and Liu, Q. (2020). SparTerm: Learning term-based sparse representation for fast text retrieval. *arXiv preprint arXiv:2010.00768*.

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of ACL*, pages 238–247.

Bendersky, M., Metzler, D., and Croft, W. B. (2010). Learning concept importance using a weighted dependence model. In *Proceedings of WSDM*, pages 31–40.

Bergum, J. K. (2021). Pretrained transformer language models for search - part 4.

Billerbeck, B. and Zobel, J. (2005). Document expansion versus query expansion for ad-hoc retrieval. In *Proceedings of ADCS*, pages 34–41.

Bolotova, V., Blinov, V., Scholer, F., Croft, W. B., and Sanderson, M. (2022). A non-factoid question-answering taxonomy. In *Proceedings of SIGIR*, pages 1196–1207.

Bondarenko, A., Fröbe, M., Beloucif, M., Gienapp, L., Ajjour, Y., Panchenko, A., Biemann, C., Stein, B., Wachsmuth, H., Potthast, M., et al. (2020). Overview of touché 2020: argument retrieval. In *Proceddings of CLEF*, pages 384–395.

Bostrom, K. and Durrett, G. (2020). Byte pair encoding is suboptimal for language model pretraining. In *Proceedings of EMNLP: Findings*, pages 4617–4624.

Boteva, V., Gholipour, D., Sokolov, A., and Riezler, S. (2016). A full-text learning to rank dataset for medical information retrieval. In *Proceddings of ECIR*, pages 716–722.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. In *Proceedings of NeurIPS*, volume 33, pages 1877–1901.

Cao, G., Nie, J.-Y., Gao, J., and Robertson, S. (2008). Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of SIGIR*, pages 243–250.

Carmel, D. and Yom-Tov, E. (2010). Estimating the query difficulty for information retrieval. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 2(1):1–89.

Chen, T., Zhang, M., Lu, J., Bendersky, M., and Najork, M. (2022). Out-of-domain semantics to the rescue! zero-shot hybrid retrieval models. In *Proceedings of ECIR*, pages 95–110.

Cho, K., Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of EMNLP*.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. (2022). Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceddings of ICLR*.

Clarke, C. L., Craswell, N., and Soboroff, I. (2009). Overview of the trec 2009 web track. Technical report, WATERLOO UNIV (ONTARIO).

Clarke, C. L., Craswell, N., Soboroff, I., et al. (2004). Overview of the trec 2004 terabyte track. In *TREC*, volume 4, page 74.

Cleverdon, C., Mills, J., and Keen, M. (1966). Factors determining the performance of indexing systems. *ASLIB Cranfield Research Project*, 50(2).

Clinchant, S. and Gaussier, E. (2011). Is document frequency important for PRF? In *Proceedings of ICTIR*, pages 89–100.

Collins-Thompson, K., Macdonald, C., Bennett, P. N., Diaz, F., and Voorhees, E. M. (2014). Trec 2014 web track overview. In *Proceedings of TREC*, volume 13, pages 1–15.

Craswell, N. and Hawking, D. (2004). Overview of the trec-2004 web track. In *Proceedings of TREC*.

Craswell, N., Hawking, D., Wilkinson, R., and Wu, M. (2003). Overview of the trec-2003 web track. In *Proceedings of TREC*.

Craswell, N., Mitra, B., ilmaz, E., Campos, D., Lin, J., Voorhees, E. M., and Soboroff, I. (2023). Overview of the TREC 2022 deep learning track. In *Proceedings of TREC*.

Craswell, N., Mitra, B., Yilmaz, E., and Campos, D. (2021a). Overview of the TREC 2020 deep learning track. In *Proceedings of TREC*.

Craswell, N., Mitra, B., Yilmaz, E., Campos, D., and Voorhees, E. M. (2020). Overview of the TREC 2019 deep learning track. In *Proceedings of TREC*.

Craswell, N., Mitra, B., Yilmaz, E., and Daniel Campos, J. L. (2021b). Overview of the TREC 2021 deep learning track. In *Proceedings of TREC*.

Craswell, N., Zoeter, O., Taylor, M., and Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proceedings of WSDM*, pages 87–94.

Croft, W. B., Metzler, D., and Strohman, T. (2010). *Search engines: Information retrieval in practice*, volume 520. Addison-Wesley Reading.

Cronen-Townsend, S., Zhou, Y., and Croft, W. B. (2004). A framework for selective query expansion. In *Proceedings of CIKM*, pages 236–237.

Dai, Z. and Callan, J. (2019a). Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of SIGIR*, pages 985–988.

Dai, Z. and Callan, J. (2019b). Deeper text understanding for IR with contextual neural language modeling. In *Proceedings of SIGIR*, page 985–988.

Dai, Z. and Callan, J. (2020a). Context-aware document term weighting for ad-hoc search. In *Proceedings of WWW*, pages 1897–1907.

Dai, Z. and Callan, J. (2020b). Context-aware passage term weighting for first stage retrieval. In *Proceedings of SIGIR*, pages 1533—-1536.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of ACL*, pages 4171–4186.

Diaz, F. and Metzler, D. (2006). Improving the estimation of relevance models using large external corpora. In *Proceedings of SIGIR*, pages 154–161.

Diaz, F., Mitra, B., and Craswell, N. (2016). Query expansion with locally-trained word embeddings. In *Proceedings of ACL*, pages 367–377.

Formal, T., Lassance, C., Piwowarski, B., and Clinchant, S. (2022). From distillation to hard negative sampling: Making sparse neural ir models more effective. In *Proceedings of SIGIR*, pages 2353—2359.

Formal, T., Piwowarski, B., and Clinchant, S. (2021a). SPLADE: Sparse lexical and expansion model for first stage ranking. In *Proceedings of SIGIR*, pages 2288–2292.

Formal, T., Piwowarski, B., and Clinchant, S. (2021b). A white box analysis of ColBERT. In *Proceedings of ECIR*, pages 257–263.

Frachtenberg, E. (2009). Reducing query latencies in web search using fine-grained parallelism. *World Wide Web*, 12(4):441.

Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. (1987). The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971.

Gao, L., Dai, Z., and Callan, J. (2021a). Coil: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of ACL*, pages 3030–3042.

Gao, L., Dai, Z., and Callan, J. (2021b). Rethink training of bert rerankers in multi-stage retrieval pipeline. In *Proceedings of ECIR*, pages 280–286.

Gao, L., Dai, Z., Chen, T., Fan, Z., Van Durme, B., and Callan, J. (2020). Complementing lexical retrieval with semantic residual embedding. In *Proceedings of ECIR*, pages 146–160.

Gospodinov, M., MacAvaney, S., and Macdonald, C. (2023). Doc2query: When less is more. In *Proceedings of ECIR*, pages 414—422.

Guo, J., Fan, Y., Ai, Q., and Croft, W. B. (2016). A deep relevance matching model for ad-hoc retrieval. In *Proceedings of CIKM*, pages 55–64.

Guo, W., Zhao, M., Zhang, L., Niu, D., Luo, J., Liu, Z., Li, Z., and Tang, J. (2021). Lichee: Improving language model pre-training with multi-grained tokenization. In *Proceedings of ACL-IJCNLP: Findings*, pages 1383–1392.

Harman, D. (1995). Overview of the second text retrieval conference (trec-2). *Information Processing & Management*, 31(3):271–289.

Hasibi, F., Nikolaev, F., Xiong, C., Balog, K., Bratsberg, S. E., Kotov, A., and Callan, J. (2017). Dbpedia-entity v2: a test collection for entity search. In *Proceedings of SIGIR*, pages 1265–1268.

He, B. and Ounis, I. (2004). Inferring query performance using pre-retrieval predictors. In *Proceedings of SPIRE*, pages 43–54.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Imani, A., Vakili, A., Montazer, A., and Shakery, A. (2019). Deep neural networks for query expansion using word embeddings. In *Proceedings of ECIR*, pages 203–210.

Järvelin, K. and Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2015). On using very large target vocabulary for neural machine translation. In *Proceedings of ACL*, pages 1–10.

Johnson, J., Douze, M., and Jégou, H. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Jones, R., Rey, B., Madani, O., and Greiner, W. (2006). Generating query substitutions. In *Proceedings of WWW*, pages 387–396.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling laws for neural language models. In *arXiv preprint arXiv:2001.08361*.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In *Proceedings of EMNLP*, pages 6769–6781.

Khattab, O. and Zaharia, M. (2020). ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of SIGIR*, pages 39–48.

Khennak, I., Drias, H., Kechid, A., and Moulai, H. (2019). Clustering algorithms for query expansion based information retrieval. In *Proceedings of ICCI*, pages 261–272.

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL*, pages 67–72.

Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP*, pages 66–71.

Kuzi, S., Shtok, A., and Kurland, O. (2016). Query expansion using word embeddings. In *Proceedings of CIKM*, pages 1929–1932.

Kwok, K. L. and Chan, M. (1998). Improving two-stage ad-hoc retrieval for short queries. In *Proceedings of SIGIR*, pages 250–256.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceddings of ICLR*.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of ICML*, pages 1188–1196.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, J., Dai, Z., Duddu, S. M. K., Lei, T., Naim, I., Chang, M.-W., and Zhao, V. Y. (2023). Rethinking the role of token retrieval in multi-vector retrieval. *arXiv preprint arXiv:2304.01982*.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, pages 7871–7880.

Li, C., Sun, Y., He, B., Wang, L., Hui, K., Yates, A., Sun, L., and Xu, J. (2018). NPRF: A neural pseudo relevance feedback framework for ad-hoc information retrieval. In *Proceedings of EMNLP*, pages 4482–4491.

Li, C., Yates, A., MacAvaney, S., He, B., and Sun, Y. (2020). Parade: Passage representation aggregation for document reranking. *ACM Transactions on Information Systems*.

Li, H., Mourad, A., Zhuang, S., Koopman, B., and Zuccon, G. (2021a). Pseudo relevance feedback with deep language models and dense retrievers: Successes and pitfalls. *ACM Transactions on Information Systems (TOIS)*.

Li, H., Zhuang, S., Mourad, A., Ma, X., Lin, J., and Zuccon, G. (2021b). Improving query representations for dense retrieval with pseudo relevance feedback: A reproducibility study. In *Proceedings of ECIR*, pages 599–612.

Lin, J. (2019). The neural hype and comparisons against weak baselines. *SIGIR Forum*, pages 40–51.

Lin, J. and Ma, X. (2021). A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. *arXiv preprint arXiv:2106.14807*.

Lin, J., Nogueira, R., and Yates, A. (2021a). Pretrained transformers for text ranking: BERT and beyond. *Synthesis Lectures on Human Language Technologies*, 14(4):1–325.

Lin, S.-C., Yang, J.-H., and Lin, J. (2020a). Distilling dense representations for ranking using tightly-coupled teachers. *arXiv preprint arXiv:2010.11386*.

Lin, S.-C., Yang, J.-H., and Lin, J. (2021b). In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of Workshop on RepL4NLP*, pages 163–173.

Lin, S.-C., Yang, J.-H., Nogueira, R., Tsai, M.-F., Wang, C.-J., and Lin, J. (2020b). Conversational question reformulation via sequence-to-sequence architectures and pretrained language models. *arXiv preprint arXiv:2004.01909*.

Lin, S.-C., Yang, J.-H., Nogueira, R., Tsai, M.-F., Wang, C.-J., and Lin, J. (2020c). Query reformulation using query history for passage retrieval in conversational search. *arXiv preprint arXiv:2005.02230*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2020). RoBERTa: A robustly optimized BERT pretraining approach. In *Proceddings of ICLR*.

Luan, Y., Eisenstein, J., Toutanova, K., and Collins, M. (2021). Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Ma, X., Pradeep, R., Nogueira, R., and Lin, J. (2022). Document expansion baselines and learned sparse lexical representations for ms marco v1 and v2. In *Proceedings of SIGIR*, pages 3187–3197.

MacAvaney, S. (2020). Opennir: A complete neural ad-hoc ranking pipeline. In *Proceedings of CIKM*, pages 845–848.

MacAvaney, S., Cohan, A., and Goharian, N. (2020a). Sledge-z: A zero-shot baseline for covid-19 literature search. In *Proceedings of EMNLP*, pages 4171–4179.

MacAvaney, S., Nardini, F. M., Perego, R., Tonellotto, N., Goharian, N., and Frieder, O. (2020b). Expansion via prediction of importance with contextualization. In *Proceedings of SIGIR*, pages 1573–1576.

MacAvaney, S., Tonellotto, N., and Macdonald, C. (2022). Adaptive re-ranking with a corpus graph. In *Proceedings of CIKM*, pages 1491–1500.

MacAvaney, S., Yates, A., Cohan, A., and Goharian, N. (2019). CEDR: Contextualized embeddings for document ranking. In *Proceedings of SIGIR*, pages 1101–1104.

Macdonald, C. and Tonellotto, N. (2020). Declarative experimentation in information retrieval using PyTerrier. In *Proceedings of ICTIR*, page 4526–4533.

Macdonald, C. and Tonellotto, N. (2021). On approximate nearest neighbour selection for multi-stage dense retrieval. In *Proceedings of CIKM*, pages 3318–3322.

Macdonald, C., Tonellotto, N., MacAvaney, S., and Ounis, I. (2021a). PyTerrier: Declarative experimentation in python from bm25 to dense retrieval. In *Proceedings of CIKM*, pages 4526–4533.

Macdonald, C., Tonellotto, N., and Ounis, I. (2021b). On single and multiple representations in dense passage retrieval. *IIR 2021 Workshop*.

Mackie, I., Chatterjee, S., and Dalton, J. (2023). Generative relevance feedback with large language models. In *Proceedings of SIGIR*.

Mallia, A., Khattab, O., Suel, T., and Tonellotto, N. (2021). Learning passage impacts for inverted indexes. In *Proceedings of SIGIR*, pages 1723–1727.

Manning, C. D. (2009). *An introduction to information retrieval*. Cambridge university press.

Mao, Y., He, P., Liu, X., Shen, Y., Gao, J., Han, J., and Chen, W. (2021). Generation-augmented retrieval for open-domain question answering. In *Proceedings of ACL*, pages 4089–4100.

Mass, Y., Carmeli, B., Roitman, H., and Konopnicki, D. (2020). Unsupervised FAQ retrieval with question generation and BERT. In *Proceedings of ACL*, pages 807–812.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceddings of NeurIPS*, volume 26.

Nair, S., Yang, E., Lawrie, D., Duh, K., McNamee, P., Murray, K., Mayfield, J., and Oard, D. W. (2022). Transfer learning approaches for building cross-language dense retrieval models. In *roceddings of ECIR*, pages 382–396.

Naseri, S., Dalton, J., Yates, A., and Allan, J. (2021). CEQE: Contextualized embeddings for query expansion. *Proceedings of ECIR*, pages 467–482.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPs*.

Nogueira, R. and Cho, K. (2019). Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*.

Nogueira, R., Jiang, Z., and Lin, J. (2020). Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*.

Nogueira, R., Lin, J., and Epistemic, A. (2019a). From doc2query to docttttquery. *Online preprint*.

Nogueira, R., Yang, W., Lin, J., and Cho, K. (2019b). Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., and Johnson, D. (2005). Terrier information retrieval platform. In *Proceedings of ECIR*, pages 517–519.

Peng, J., He, B., and Ounis, I. (2009a). Predicting the usefulness of collection enrichment for enterprise search. In *Proceedings of CIKM*, pages 366–370.

Peng, J., Macdonald, C., He, B., and Ounis, I. (2009b). A study of selective collection enrichment for enterprise search. In *Proceedings of CIKM*, pages 1999–2002.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Pradeep, R., Liu, Y., Zhang, X., Li, Y., Yates, A., and Lin, J. (2022). Squeezing water from a stone: A bag of tricks for further improving cross-encoder effectiveness for reranking. In *Proceedings of ECIR*, pages 655–670.

Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W. X., Dong, D., Wu, H., and Wang, H. (2020). RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of NAACL*, pages 5835–5847.

Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W. X., Dong, D., Wu, H., and Wang, H. (2021). Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of NAACL*, pages 5835–5847.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. *OpenAI blog*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Radlinski, F. and Craswell, N. (2010). Comparing the sensitivity of information retrieval metrics. In *Proceedings of SIGIR*, pages 667–674.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, pages 1–67.

Rajab, J. (2022). Effect of tokenisation strategies for low-resourced southern african languages. In *Proceedings of Workshop on AfricaNLP*.

Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. (2007). Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, pages 464–471.

Robertson, S. (2008). On the optimisation of evaluation metrics. In *Keynote, SIGIR 2008 workshop learning to rank for information retrieval (LR4IR)*.

Robertson, S. E. (1977). The probability ranking principle in ir. *Journal of documentation*.

Robertson, S. E. and Spärck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al. (1995). Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Rocchio, J. (1971). Relevance feedback in information retrieval. *The Smart Retrieval System-experiments in Automatic Document Processing*, pages 313–323.

Roy, D., Bhatia, S., and Mitra, M. (2019). Selecting discriminative terms for relevance model. In *Proceedings of SIGIR*, pages 1253–1256.

Roy, D., Ganguly, D., Bhatia, S., Bedathur, S., and Mitra, M. (2018). Using word embeddings for information retrieval: How collection and term normalization choices affect performance. In *Proceedings of CIKM*, pages 1835–1838.

Roy, D., Paul, D., Mitra, M., and Garain, U. (2016). Using word embeddings for automatic query expansion. In *Proceedings of SIGIR Workshop on Neural Information Retrieval*. arXiv:1606.07608.

Sakai, T. (2021). On fuhr's guideline for ir evaluation. *SIGIR Forum*, 54(1).

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Sanderson, M., Paramita, M. L., Clough, P., and Kanoulas, E. (2010). Do user preferences and evaluation measures line up? In *Proceedings of SIGIR*, pages 555–562.

Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., and Zaharia, M. (2022). ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of NAACL*, pages 3715–3734.

Scells, H., Zhuang, S., and Zuccon, G. (2022). Reduce, reuse, recycle: Green information retrieval research. In *Proceedings of SIGIR*, pages 2825–2837.

Scholer, F., Williams, H. E., Yiannis, J., and Zobel, J. (2002). Compression of inverted indexes for fast query evaluation. In *Proceedings of SIGIR*, pages 222–229.

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *Proceddings of ICASSP*, pages 5149–5152.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pages 1715–1725.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Spärck Jones, K., Walker, S., and Robertson, S. E. (2000). A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Information processing & management*, 36(6):809–840.

Su, T., Wang, X., Macdonald, C., and Ounis, I. (2019). University of glasgow terrier team at the trec 2019 deep learning track. In *Proceedings of TREC*.

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I. (2021). BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of NeurIPS*.

Tonellotto, N. and Macdonald, C. (2021a). Query embedding pruning for dense retrieval. In *Proceedings of CIKM*, pages 3453–3457.

Tonellotto, N. and Macdonald, C. (2021b). Query embedding pruning for dense retrieval. In *Proceedings of CIKM*, page 3453–3457.

Tonellotto, N., Macdonald, C., and Ounis, I. (2013). Efficient and effective retrieval using selective pruning. In *Proceedings of WSDM*, pages 63–72.

Tonellotto, N., Macdonald, C., Ounis, I., et al. (2018). Efficient query processing for scalable web search. *Foundations and Trends® in Information Retrieval*, 12(4-5):319–500.

Toraman, C., Yilmaz, E. H., Şahinuç, F., and Ozcelik, O. (2022). Impact of tokenization on language models: An analysis for turkish. *arXiv preprint arXiv:2204.08832*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of NeurIPS*, pages 5998–6008.

Voorhees, E. (2005). The TREC robust retrieval track. In *ACM SIGIR Forum*, volume 39, pages 11–20.

Voorhees, E. (2006). The TREC 2005 robust track. In *ACM SIGIR Forum*, volume 40, pages 41–48.

Voorhees, E., Alam, T., Bedrick, S., Demner-Fushman, D., Hersh, W. R., Lo, K., Roberts, K., Soboroff, I., and Wang, L. L. (2021). TREC-COVID: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, pages 1–12.

Voorhees, E. M. (2004). Overview of the trec 2004 robust track,. In *Proceedings of TREC*.

Wang, L., Yang, N., and Wei, F. (2023a). Query2doc: Query expansion with large language models. *arXiv preprint arXiv:2303.07678*.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. (2020a). MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceddings of NeurIPS*, volume 33, pages 5776–5788.

Wang, X., MacAvaney, S., Macdonald, C., and Ounis, I. (2022a). An inspection of the reproducibility and replicability of TCT-ColBERT. In *Proceedings of SIGIR*, pages 2790–2800.

Wang, X., MacAvaney, S., Macdonald, C., and Ounis, I. (2023b). Effective contrastive weighting for dense query expansion. In *Proceedings of ACL*, pages 12688–12704.

Wang, X., MacAvaney, S., Macdonald, C., and Ounis, I. (2023c). Generative Query Reformulation for Effective Adhoc Retrieval . In *Proceedings of Workshop on GenIR*.

Wang, X., Macdonald, C., and Ounis, I. (2020b). Deep reinforced query reformulation for information retrieval. *arXiv preprint arXiv:2007.07987*.

Wang, X., Macdonald, C., and Ounis, I. (2022b). Improving zero-shot retrieval using dense external expansion. *Information Processing & Management*, 59(5):103026.

Wang, X., Macdonald, C., Tonellotto, N., and Ounis, I. (2021a). Pseudo-relevance feedback for multiple representation dense retrieval. In *Proceedings of ICTIR*, pages 297–306.

Wang, X., Macdonald, C., Tonellotto, N., and Ounis, I. (2022c). ColBERT-PRF: Semantic pseudo-relevance feedback for dense passage and document retrieval. *ACM Transactions on the Web*, 17:1–39.

Wang, X., Macdonald, C., Tonellotto, N., and Ounis, I. (2023d). Reproducibility, Replicability, and Insights into Dense Multi-Representation Retrieval Models: from ColBERT to Col⋆. In *Proceedings of SIGIR*.

Wang, Y., Li, J., Naumann, T., Xiong, C., Cheng, H., Tinn, R., Wong, C., Usuyama, N., Rogahn, R., Shen, Z., Qin, Y., Horvitz, E., Bennett, P., Gao, J., and Poon, H. (2021b). Domain-specific pretraining for vertical search: Case study on biomedical literature. *Proceedings of SIGKDD*, pages 3717–3725.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2021). Finetuned language models are zero-shot learners. In *Proceedings of ICLR*.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. (2022a). Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022b). Chain-of-thought prompting elicits reasoning in large language models. In *Proceddings of NeurIPS*, volume 35, pages 24824–24837.

Xiong, C., Dai, Z., Callan, J., Liu, Z., and Power, R. (2017). End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of SIGIR*, pages 55–64.

Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J., and Overwijk, A. (2021). Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of ICLR*.

Xu, J. and Croft, W. B. (2017). Quary expansion using local and global document analysis. *Acm sigir forum*, 51(2):168–175.

Xu, Y., Jones, G. J., and Wang, B. (2009). Query dependent pseudo-relevance feedback based on Wikipedia. In *Proceedings of SIGIR*, pages 59–66.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceddings of NeurIPS*, volume 32.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Yih, W.-t., Toutanova, K., Platt, J. C., and Meek, C. (2011). Learning discriminative projections for text similarity measures. In *Proceedings of ACL*, pages 247–256.

Yu, H., Dai, Z., and Callan, J. (2021a). PGT: pseudo relevance feedback using a graph-based transformer. In *Proceedings of ECIR*, pages 440–447.

Yu, H., Xiong, C., and Callan, J. (2021b). Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of CIKM*, pages 599–612.

Yu, S., Liu, J., Yang, J., Xiong, C., Bennett, P., Gao, J., and Liu, Z. (2020). Few-shot generative conversational query rewriting. In *Proceedings of SIGIR*, pages 1933–1936.

Zamani, H. and Croft, W. B. (2016). Embedding-based query language models. In *Proceedings of ICTIR*, pages 147–156.

Zamani, H. and Croft, W. B. (2017). Relevance-based word embedding. In *Proceedings of SIGIR*, pages 505–514.

Zerveas, G., Zhang, R., Kim, L., and Eickhoff, C. (2019). Brown University at TREC deep learning 2019. In *Proceedings of TREC*.

Zhan, J., Mao, J., Liu, Y., Guo, J., Zhang, M., and Ma, S. (2021). Optimizing dense retrieval model training with hard negatives. In *Proceedings of SIGIR*, pages 1503–1512.

Zhan, J., Mao, J., Liu, Y., Zhang, M., and Ma, S. (2020). RepBERT: Contextualized text embeddings for first-stage retrieval. *arXiv preprint arXiv:2006.15498*.

Zheng, Z., Hui, K., He, B., Han, X., Sun, L., and Yates, A. (2020). BERT-QE: Contextualized query expansion for document re-ranking. In *Proceedings of EMNLP: Findings*, pages 4718–4728.

Zhong, R., Ghosh, D., Klein, D., and Steinhardt, J. (2021). Are larger pretrained language models uniformly better? comparing performance at the instance level. In *Proceedings of ACL: Findings*, pages 3813–3827.

Zhuang, H., Qin, Z., Jagerman, R., Hui, K., Ma, J., Lu, J., Ni, J., Wang, X., and Bendersky, M. (2022). RankT5: Fine-tuning T5 for text ranking with ranking losses. *arXiv preprint arXiv:2210.10634*.

Zukerman, I. and Raskutti, B. (2002). Lexical query paraphrasing for document retrieval. In *Proceedings of COLING*, pages 1–7.

Appendix A

Prompts

For the FlanQR model, to determine the prompt input into the FLAN model to perform the query reformulation task in Section 3.3.2, we examined 14 different prompts. Table A.1 lists the prompts as well as their corresponding retrieval effectiveness, in terms of nDCG@10, on the TREC 2019 query set.

Table A.1: Prompts for the FlanQR model (with $\beta = 1$). Retrieval effectiveness is evaluated in terms of nDCG@10 on the TREC 2019 query set. The prompt with the highest retrieval effectiveness is highlighted in bold.

Prompts	nDCG@10
Provide 10 related keywords for the query: input query	0.4480
What are some synonyms for the keywords in my query that might help me refine my search, query: input query	0.4024
Expand the following query with relevant terms: input query	0.5169
Provide additional keywords to better represent the information need in the query: input query	0.4611
Suggest related terms to enhance the search query: input query	0.4389
What are some related phrases or words to refine the query: input query	0.4970
Improve the search effectiveness by suggesting expansion terms for the query: input query	0.4884
Improve the search effectiveness by suggesting 10 expansion terms for the query: input query	0.4912
List relevant terms to augment the given query for better search results: input query	0.5026
For the query: input query, identify associated words or phrases to improve information retrieval	0.4635
Help refine the search query: input query, by providing semantically related terms	0.4207
To better capture the users information need, suggest expansion terms for the query: input query	**0.5252**
To better capture the users information need, suggest 10 expansion terms for the query: input query	0.4711
Clarify the query: input query by generating contextually relevant words or phrases.	0.4611