ColBERT-PRF: Semantic Pseudo-Relevance Feedback for Dense Passage and Document Retrieval[*]

XIAO WANG, University of Glasgow, UK
CRAIG MACDONALD, University of Glasgow, UK
NICOLA TONELLOTTO, University of Pisa, Italy
IADH OUNIS, University of Glasgow, UK

Pseudo-relevance feedback mechanisms, from Rocchio to the relevance models, have shown the usefulness of expanding and reweighting the users' initial queries using information occurring in an initial set of retrieved documents, known as the pseudo-relevant set. Recently, dense retrieval – through the use of neural contextual language models such as BERT for analysing the documents' and queries' contents and computing their relevance scores – has shown a promising performance on several information retrieval tasks still relying on the traditional inverted index for identifying documents relevant to a query. Two different dense retrieval families have emerged: the use of single embedded representations for each passage and query, e.g., using BERT's [CLS] token, or via multiple representations, e.g., using an embedding for each token of the query and document (exemplified by ColBERT). In this work, we conduct the first study into the potential for multiple representation dense retrieval to be enhanced using pseudo-relevance feedback and present our proposed approach ColBERT-PRF. In particular, based on the pseudo-relevant set of documents identified using a first-pass dense retrieval, ColBERT-PRF extracts the representative feedback embeddings from the document embeddings of the pseudo-relevant set. Among the representative feedback embeddings, the embeddings that most highly discriminate among documents are employed as the expansion embeddings, which are then added to the original query representation. We show that these additional expansion embeddings both enhance the effectiveness of a reranking of the initial query results as well as an additional dense retrieval operation. Indeed, experiments on the MSMARCO passage ranking dataset show that MAP can be improved by upto 26% on the TREC 2019 query set and 10% on the TREC 2020 query set by the application of our proposed ColBERT-PRF method on a ColBERT dense retrieval approach. We further validate the effectiveness of our proposed pseudo-relevance feedback technique for a dense retrieval model on MSMARCO document ranking and TREC Robust04 document ranking tasks. For instance, ColBERT-PRF exhibits upto 21% and 14% improvement in MAP over the ColBERT E2E model on the MSMARCO document ranking TREC 2019 and TREC 2020 query sets, respectively. Additionally, 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, attaining upto 4.54x speedup over the default ColBERT-PRF model, and with little impact on effectiveness, through the application of approximate scoring and different clustering methods.

CCS Concepts: • **Information systems → Information retrieval query processing**; **Information retrieval**.

Additional Key Words and Phrases: Query Expansion; Pseudo-Relevance Feedback; BERT; Dense Retrieval

ACM Reference Format:
Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2022. ColBERT-PRF: Semantic Pseudo-Relevance Feedback for Dense Passage and Document Retrieval. *ACM Trans. Web* 1, 1 (September 2022), 40 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[*]This manuscript extends an earlier ICTIR 2021 publication [44].

Authors' addresses: Xiao Wang, University of Glasgow, UK, x.wang.8@research.gla.ac.uk; Craig Macdonald, University of Glasgow, UK, firstname.lastname@glasgow.ac.uk; Nicola Tonellotto, University of Pisa, Italy, firstname.lastname@unipi.it; Iadh Ounis, University of Glasgow, UK, firstname.lastname@glasgow.ac.uk.

1 INTRODUCTION

When searching for information, users often formulate queries in a different way to the relevant documents. For instance, a user may search for information about "surname meaning" using a query "where do last names come from". However, a relevant document may describe the "last name" using "family name" or "surname" and may use terms such as "originate" or "history" instead of "come from". Thus, a relevant document and the user query might form a lexical mismatch gap during retrieval, which must be bridged for effective retrieval.

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. Many approaches follow the pseudo-relevance feedback (PRF) paradigm – such as Rocchio's algorithm [37], the RM3 relevance language model [1], or the DFR query expansion models [4] – where terms appearing in the top-ranked documents for the initial query are used to expand it. Query expansion (QE) approaches have also found a useful role 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 [35, 42, 47].

On the other hand, many studies have focused on the use of *static* word embeddings, such as Word2Vec, within query expansion methods [12, 19, 39, 40]. Indeed, most of the existing embedding-based QE methods [12, 19, 39, 40, 49] 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. Recently, CEQE [29] was proposed, which makes use of contextualised BERT embeddings for query expansion. The resulting refined query representation is then used for a further round of retrieval using a traditional (sparse) inverted index. In contrast, in this paper, we focus on implementing contextualised embedding-based query expansion for dense retrieval.

Indeed, the BERT models have demonstrated further promise in being a suitable basis for *dense retrieval.* In particular, 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 [15]. Two distinct families of approaches have emerged: single representation dense retrieval and multiple representation dense retrieval. In single representation dense retrieval, as used by DPR [16] and ANCE [46], each query or document is represented entirely by the single embedding of the [CLS] (classification) token computed by BERT. Query-document relevance is estimated in terms of the similarity of the corresponding [CLS] embeddings. In contrast, in multiple representation dense retrieval – as proposed by ColBERT [17] – 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 the contextualised representation 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" [36]. We argue that, as users issue the query prior to the 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, in this work, we are concerned with applying pseudo-relevance feedback in a multiple representation dense retrieval setting. 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.

Thus, in this work, we propose a pseudo-relevance feedback mechanism called ColBERT-PRF for dense retrieval. In particular, 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. In summary, a preliminary version of this paper appeared in ICTIR 2021 [44] which made the following contributions: (1) we propose a novel contextualised pseudo-relevance feedback mechanism for multiple representation dense retrieval; (2) we cluster and rank the feedback document embeddings for selecting candidate expansion embeddings; (3) we evaluate our proposed contextualised PRF model in both ranking and reranking settings. In this work, we extend our previous work and thus make the following additional contributions: (4) we demonstrate the effectiveness of ColBERT-PRF model on document ranking tasks, using the MSMARCO document test collection and the TREC Robust04 test collections; (5) We further investigate the effectiveness of ColBERT-PRF by varying the selection of the expansion embeddings. (6) We thoroughly investigate the trade-off between the effectiveness and the efficiency of ColBERT-PRF.

The remainder of this paper is as follows: Section 2 positions this work among existing approaches to pseudo-relevance feedback; Section 3 describes a multi-representation dense retrieval, while Section 4 presents our proposed dense PRF method. Next, we discuss the effectiveness of ColBERT-PRF for passage ranking task and for document ranking task in Section 5 and Section 6, respectively. Next, we discuss the usefulness of different weighting methods for measuring the informativeness of the expansion embeddings of ColBERT-PRF in Section 7. In Section 8, we study efficient variants of ColBERT-PRF. Finally, we provide concluding remarks and a discussion of future directions in Section 9.

2 RELATED WORK

Pseudo-relevance feedback approaches have a long history in Information Retrieval (IR) going back to Rocchio [37] who generated refined query reformulations through linear combinations of the sparse vectors, e.g., containing term frequency information representing the query and the top-ranked feedback documents. Refined classical PRF models, such as Divergence from Randomness's Bo1 [4], KL [2], and RM3 relevance models [1] have demonstrated their effectiveness on many test collections. Typically, these models identify and weight feedback terms that are frequent in the feedback documents and infrequent in the corpus, by exploiting statistical information about the occurrence of terms in the documents and in the whole collection. In all cases, the reformulated query is then re-executed on the traditional (so-called *sparse*) inverted index.

Recently, deep learning solutions based on transformer networks have been used to enrich the statistical information about terms by rewriting or expanding the collection of documents. For instance, DeepCT [10] reweights terms occurring in the documents according to a fine-tuned BERT model to highlight important terms. This results in *augmented* document representations, which can be indexed using a traditional inverted indexer. Similarly, doc2query [33] and its more modern variant docT5query [32] apply text-to-text translation models to each document in the collection to suggest queries that may be relevant to the document. When the suggested queries are indexed along with the original document, the retrieval effectiveness is enhanced.

More recently, instead of leveraging (augmented) statistical information such as the in-document and collection frequency of terms to model a query or a document, dense representations, also known as embeddings, are becoming commonplace. Embeddings encode terms in queries and documents by learning a vector representation for each term, which takes into account the word semantic and context. Instead of identifying the related terms in the pseudo-relevance feedback documents using statistical methods, embedding-based query expansion methods [12, 19, 39, 40, 49] expand a query with terms that are closest to the query terms in the word embedding space. However, the expansion terms may not be sufficiently informative to distinguish relevant documents from non-relevant documents – for instance, the embedding of "grows" may be closest to "grow" in the embedding space, but adding "grows" to the query may not help to identify more relevant documents. Moreover, all these embedding-based method are based on non-contextualised embeddings, where a word embedding is always the same within different sentences, and hence they do not address contextualised language models. Pre-trained contextualised language models such as BERT [11] have brought large effectiveness improvements over prior art in information retrieval tasks. In particular, deep learning is able to successfully exploit general language features in order to capture the contextual semantic signals allowing to better estimate the relevance of documents w.r.t. a given query.

Query expansion approaches have been used for generating a high quality pool of candidate documents to be reranked by effective BERT-based neural reranking models [35, 42, 47]. However the use of BERT models directly within the pseudo-relevance feedback mechanism has seen comparatively little use in the literature. The current approaches leveraging the BERT contextualised embeddings for PRF are Neural PRF [20], BERT-QE [51] and CEQE [29].

In particular, Neural PRF uses neural ranking models, such as DRMM [14] and KNRM [45], to score the similarity of a document to a top-ranked feedback document. BERT-QE is conceptually similar to Neural PRF, but it measures the similarity of each document w.r.t. feedback chunks that are extracted from the top-ranked feedback documents. This results in an expensive application of many BERT computations – approximately 11× as many GPU operations than a simple BERT reranker [51]. 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.

Meanwhile, Rocchio's relevance feedback algorithm has also been implemented for a learned sparse index by SNRM [50]. However, this model relies on a sparse index representation, which looses the advantages of dense retrieval. CEQE exploits BERT to compute contextualised representations for the query as well as for the terms in the top-ranked feedback documents, and then selects as expansion terms those which are the closest to the query embeddings according to some similarity measure. In contrast to Neural PRF and BERT-QE, CEQE is used to generate a new query of terms for execution upon a traditional (sparse) inverted index. 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.

In contrast to the aforementioned approaches, our proposed ColBERT-PRF approach can be exploited in a dense retrieval system, both in end-to-end ranking and reranking scenarios. Dense retrieval approaches, exemplified by ANCE [46] and ColBERT [17], are of increasing interest, due to their use of the BERT embedding(s) for representing queries and documents. By using directly the BERT embeddings for retrieval, topic drifts for polysemous words can be avoided. Concurrently to our work, ANCE-PRF [22, 48] has been proposed to improve the effectiveness for a single representation ANCE model by retraining the query encoder using pseudo-relevance feedback information. In contrast, our work doesn't require any further training. To the best of our knowledge, ColBERT-PRF is the first work investigating PRF for a multiple representation dense retrieval setting.

3 MULTI REPRESENTATION DENSE RETRIEVAL

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 & Zaharia [17] 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 4.4, 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.

The similarity of two embeddings is computed by the dot product. Hence, for a query q and a document d, their similarity score $s(q, d)$ is obtained by summing the maximum similarity between the query token embeddings and the document token embeddings [17]:

$$s(q,d) = \sum_{i=1}^{|q|} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j} \tag{1}$$

Indeed, Formal et al. [13] showed that the dot product $\phi_{q_i}^T \phi_{d_j}$ used by ColBERT implicitly encapsulates token importance, by giving higher scores to tokens that have higher IDF values.

To obtain a first set of candidate documents, Khattab & Zaharia [17] make use of FAISS, an approximate nearest neighbour search library, on the pre-computed document embeddings. Conceptually, FAISS allows to retrieve the k' documents containing the nearest neighbour 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 (1) 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 (1). 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 [17] 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 approach proposed by Khattab & Zaharia, while the top part of Table 1 summarises the notation for the main components of the algorithm.

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

[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 8.

Table 1. Summary of notation – top group for ColBERT dense retrieval; bottom group for ColBERT-PRF.

Symbol	Meaning
ϕ_{q_i}, ϕ_{d_j}	An embedding for a query token q_i or a document token d_j
$\mathcal{F}_d(\phi_{q_i}, k')$	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
v_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
$\mathcal{F}_t(v_i, r)$	Function returning the r token ids corresponding to the r closest document embeddings to embedding v_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

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.

Algorithm 1: The ColBERT E2E algorithm

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

4 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 such as Bo1 [4] and RM3 [1], 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] The bottom part of Table 1 summarises the main notations that we use in describing ColBERT-PRF.

In this section, we detail how we identify representative (centroid) embeddings from the feedback documents (Section 4.1), how we ensure that those centroid embeddings are sufficiently discriminative (Section 4.2), and how we apply these discriminative representative centroid embeddings for (re)ranking (Section 4.3). We conclude with an illustrative example (Section 4.4) and a discussion of the novelty of ColBERT-PRF (Section 4.5).

4.1 Representative Embeddings in Feedback Documents

First, we need to identify representative embeddings $\{v_1, \dots, v_K\}$ 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 $\Phi(q, f_b)$:

$$\{v_1, .., v_K\} = \mathsf{Clustering}\big(K, \Phi(q, f_b)\big). \tag{2}$$

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. 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 query expansion methods, namely Bo1 and RM3 techniques in Appendix A.1. Later, in Section 8, we propose and evaluate other approaches for clustering. Next, we determine how well these centroids discriminate among the documents in the corpus.

4.2 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 [6, 38]. However, there may not be a one-to-one relationship between query/centroid embeddings and actual tokens, hence we seek to map each centroid v_i to a possible token t.

We resort to FAISS to achieve this, through the function $\mathcal{F}_t(v_i, r) \rightarrow (t, \dots)$ that, given the centroid embedding v_i and r, returns the list of the r *token ids* corresponding to the r closest document

[2]In Appendix A.1, we provide experiments that use Bo1 and RM3 to select tokens and their corresponding embeddings that verify this conjecture.

embeddings to the centroid.[3] From a probabilistic viewpoint, the likelihood $P(t|v_i)$ of a token t given an embedding v_i can be obtained as:

$$P(t|v_i) = \frac{1}{r} \sum_{\tau \in \mathcal{F}_t(v_i, r)} \mathbb{1}[\tau = t], \tag{3}$$

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

For simplicity, we choose the most likely token id, i.e., $t_i = \arg\max_t P(t|v_i)$. 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 v_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. In addition, we will discuss different derivations of a tailored informativeness measure in Section 7, 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:

$$F_e = \text{TopScoring}\Big(\big\{(v_1, \sigma_1), \ldots, (v_K, \sigma_K)\big\}, f_e\Big) \tag{4}$$

where TopScoring(A, c) returns the c elements of A with the highest importance score.

4.3 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. (1) as follows:

$$s(q, d) = \sum_{i=1}^{|q|} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j} + \beta \sum_{(v_i, \sigma_i) \in F_e} \max_{j=1,\ldots,|d|} \sigma_i v_i^T \phi_{d_j}, \tag{5}$$

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 (5) 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 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 max-sim scoring in Eq. (1) to encapsulate the original query embeddings as well as the expansion embeddings.

4.4 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 2(a) shows the embeddings of the original query (black

[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.

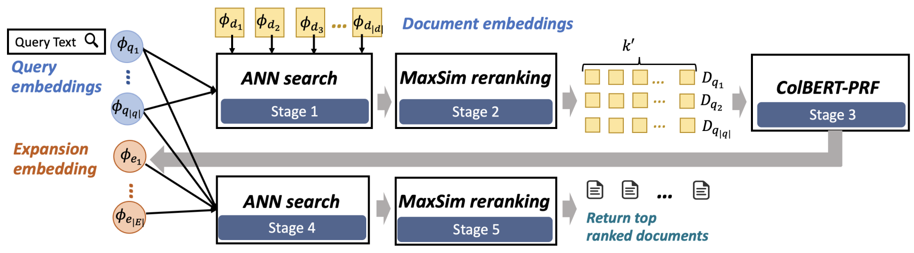
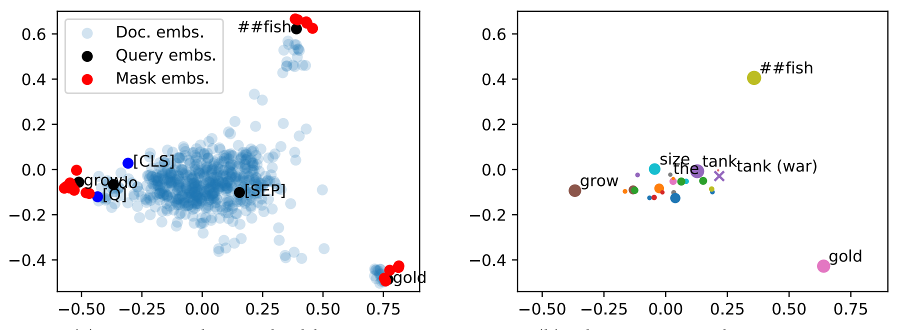


Fig. 1. Workflow of ColBERT-PRF ranker.

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($Q$):
1   $A \leftarrow$ ColBERT E2E($Q$)
2   $\Phi(Q, f_b) \leftarrow$ set of all document embeddings from
        the $f_b$ top-scored documents in $A$
3   $V \leftarrow \emptyset$
4   $v_1, \ldots, v_K$ = KMeans$(K, \Phi(Q, f_b))$
5   for $v_i$ in $v_1, \ldots, v_K$ do
6       $t_i \leftarrow \arg\max_t \frac{1}{r} \sum_{\tau \in \mathcal{F}_t(v_i, r)} \mathbb{1}[\tau = t]$
7       $\sigma_i \leftarrow \log\left(\frac{N+1}{N_i+1}\right)$
8       $V \leftarrow V \cup \{(v_i, \sigma_i)\}$
9   $F_e \leftarrow$TopScoring$(V, f_e)$
10  $B \leftarrow \emptyset$
11  for $(d, s)$ in $A$ do
12      $s \leftarrow s + \beta \sum_{(v_i, \sigma_i) \in F_e} \max_{j=1,\ldots,|d|} \sigma_i v_i^T \phi_{d_j}$
13      $B \leftarrow B \cup \{(d, s)\}$
14  return $B$
```

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 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 2(b) demonstrates the application of KMeans clustering upon the document embeddings; we map back to the original tokens by virtue of Equation (3). In Figure 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



(a) Query & doc. embeddings. (b) Cluster centroids, $K = 24$.

Fig. 2. Example showing how ColBERT-PRF operates for the query 'do goldfish grow' in a 2D PCA space. In Figure 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.

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 2(b) also includes, marked by an × and denoted 'tank (war)', the 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.

4.5 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, such as Neural PRF [20] and BERT-QE [51] only function as rerankers. Other approaches such as DeepCT [10] and doc2query [32, 33] use neural models to augment documents before indexing using a traditional inverted index. CEQE [29] 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 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 2), and their contribution to document scoring can be examined, as we will show in Section 5.3.4.

In the following, we first show the retrieval effectiveness of ColBERT-PRF for passage ranking and document ranking tasks in Section 5 and Section 6, respectively. In particular, in Section 5, 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 7, and we address the effectiveness and efficiency trade-off of ColBERT-PRF in Section 8.

5 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 5.1 describes the research question addressed by our passage ranking experiments. The experimental setup and the obtained results are detailed in Section 5.2 and Section 5.3, respectively.

5.1 Research Questions

Our passage ranking experiments address the three following research questions:

- RQ1: Can a multiple representation dense retrieval approach be enhanced by pseudo-relevance feedback, i.e., can ColBERT-PRF outperform ColBERT dense retrieval?
- RQ2: 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?
- RQ3: 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: To what extent does ColBERT-PRF perform semantic matching?

5.2 Experimental Setup

5.2.1 Dataset & Measures. Experiments are conducted on the MSMARCO passage corpus, using the 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 TRECDL passage ranking task. We omit topics from the MSMARCO Dev set, which have only sparse judgements, ~1.1 per query. Indeed, pseudo-relevance feedback approaches are known to be not effective on test collections with few judged passages [3].

We report the commonly used metrics for the TREC 2019 and TREC 2020 query sets following the corresponding track overview papers [7, 8]: we report mean reciprocal rank (MRR) and normalised discounted cumulative gain (NDCG) calculated at rank 10, as well as Recall and Mean Average Precision (MAP) at rank 1000 [8]. For the MRR, MAP and Recall metrics, we treat passages with label grade 1 as non-relevant, following [7, 8]. In addition, we also 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.

5.2.2 Implementation and Settings. We conduct experiments using PyTerrier [25] and, in particular using our PyTerrier_ColBERT plugin[5], which includes ColBERT-PRF as well as our adaptations of the ColBERT source code. ColBERT and ColBERT-PRF are expressed as PyTerrier transformer

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

operations - the source code of the ColBERF-PRF ranker and re-ranker pipelines is shown in the Appendix A.2.

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 [17]: 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 [5] of sci-kit learn (sklearn). For query expansion settings, we follow the default settings of Terrier [34], 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[6], and $\beta = \{0.5, 1\}$ for the weight of the expansion embeddings. We later show the impact of these parameters when we address RQ3.

5.2.3 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 [24] and `capreolus/ bert-base-msmarco` fine-tuned model from [21]. 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, 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: (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: The ANCE [46] 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: The ANCE-PRF [48] is a PRF variant of ANCE model – we use the results released by the authors. (iii) ColBERT E2E: ColBERT end-to-end (E2E) [17] is the dense retrieval version of ColBERT, as defined in Section 3.
BERT-QE Models: We apply BERT-QE [51] 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 [51], 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: This family consists of three CEQE variants [29], 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.

[6]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.

For reproducibility, ColBERT-PRF and the baselines results are available in our virtual appendix[7].

5.3 Passage Ranking Results

5.3.1 Results for RQ1 - 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. On analysing Table 2, 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 [23]. 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[8] 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 anker (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 the dense index. As can be seen from Table 2, 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 which can better rank documents in the existing candidate set. Overall, in response to RQ1, we conclude that our proposed ColBERT-PRF model is effective compared to the ColBERT E2E dense retrieval model.

[7]https://github.com/Xiao0728/ColBERT-PRF-VirtualAppendix
[8]Indeed, this is 8% higher than the highest MAP among all TREC 2019 participants [8].

Table 2. 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

Table 3. 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

5.3.2 *Results for RQ2 - Comparison to Baselines.*

Next, to address RQ2(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 RQ2(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 RQ2(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 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 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 3, 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 3, 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 3, 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 3 we see that ColBERT-PRF improves more queries and with further margin than ANCE-PRF.

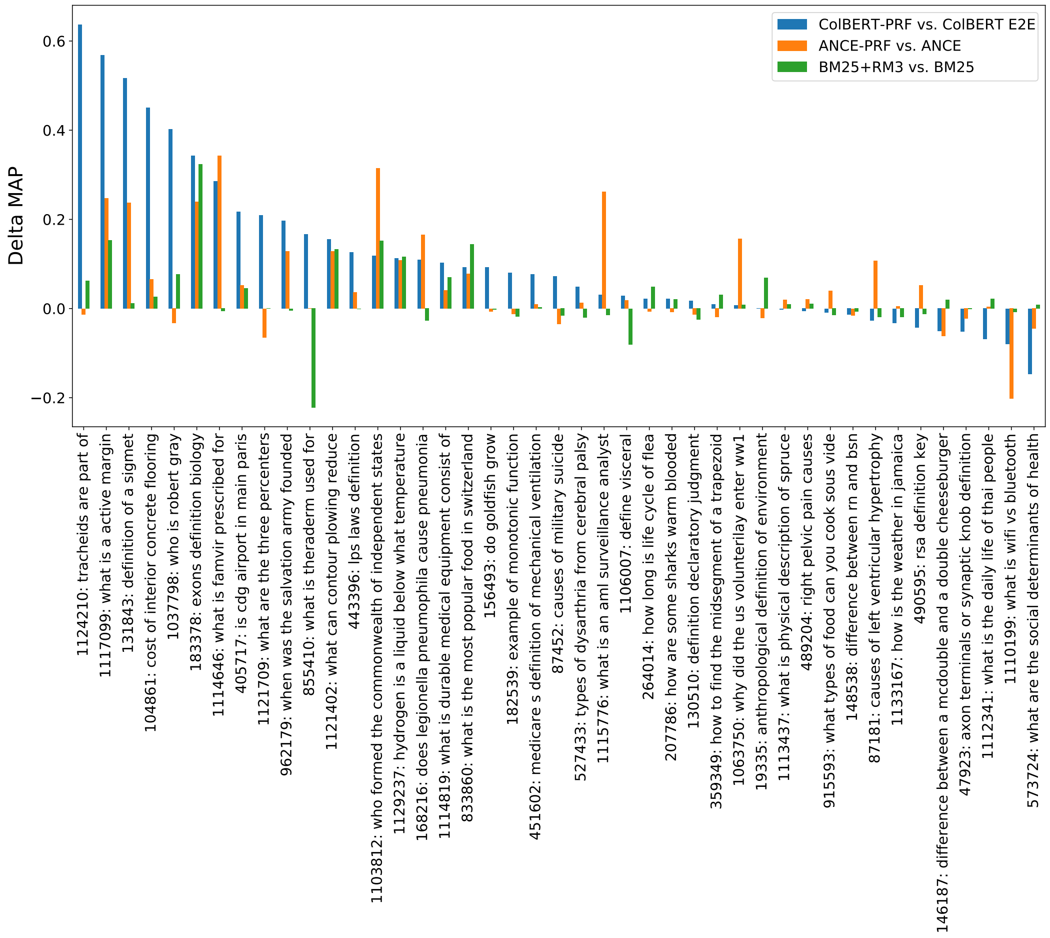
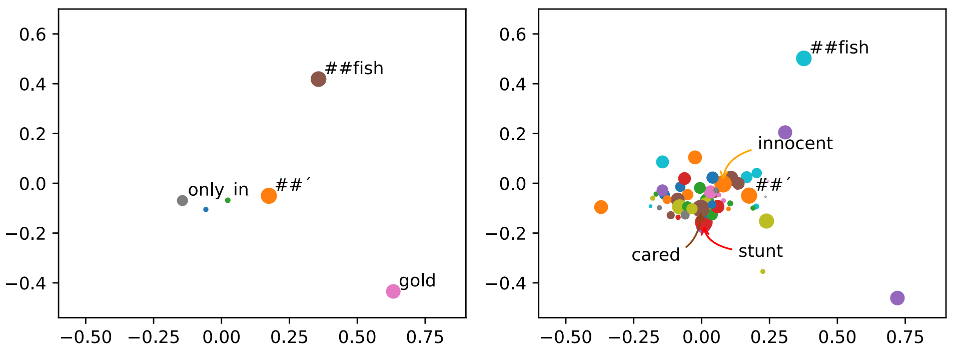


Fig. 3. Per-query analysis on the TREC 2019 query set.

For RQ2(b), on analysing the neural augmentation approaches, we observe that both the 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 RQ2(b), the ColBERT-PRF models exhibit markedly higher performances than the neural augmentation approaches.



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

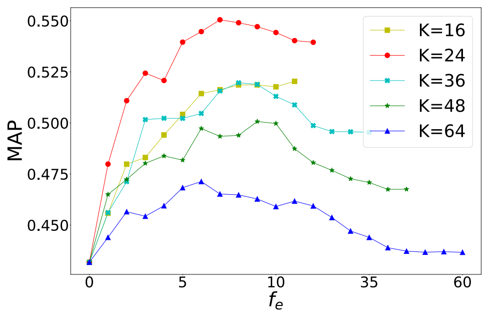
Fig. 4. Embeddings selected using different number of clustering centroids K for the query ‘do goldfish grow’; point size is representative of the magnitude of IDF.

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 RQ2(c), we find that the ColBERT-PRF models significantly outperform the BERT-QE reranking models.

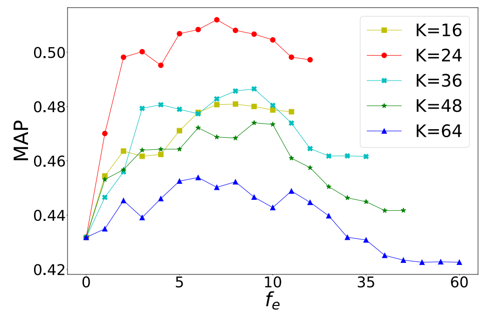
Finally, we consider the mean response times reported in Table 2, noting that ColBERT PRF exhibits higher response times than other ColBERT-based baselines, and similar to BERT-based re-rankers. There are several reasons for ColBERT PRF’s speed: Firstly, the KMeans clustering of the feedback embeddings 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 [5], 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 [26] and other clustering algorithms.

5.3.3 Results for RQ3 - Impact of ColBERT-PRF Parameters. To address RQ3, 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 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 (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 5.3.1.

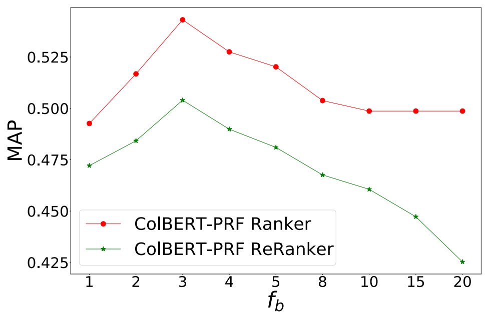
Then, for a given f_e value, Figures 5(a) and (b) show that the best performance is achieved by ColBERT-PRF when using $K = 24$. To explain this, we refer to Figure 4 together with Figure 2(b),



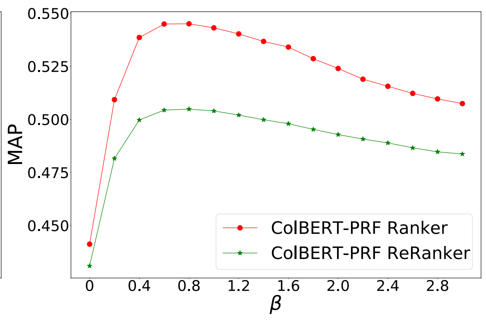
(a) Impact of K and f_e on ColBERT-PRF Ranker.



(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.

Fig. 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.

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(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(b), which could cause a topic drift.

Next, we analyse the impact of the number of feedback passages, f_b. Figure 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 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.

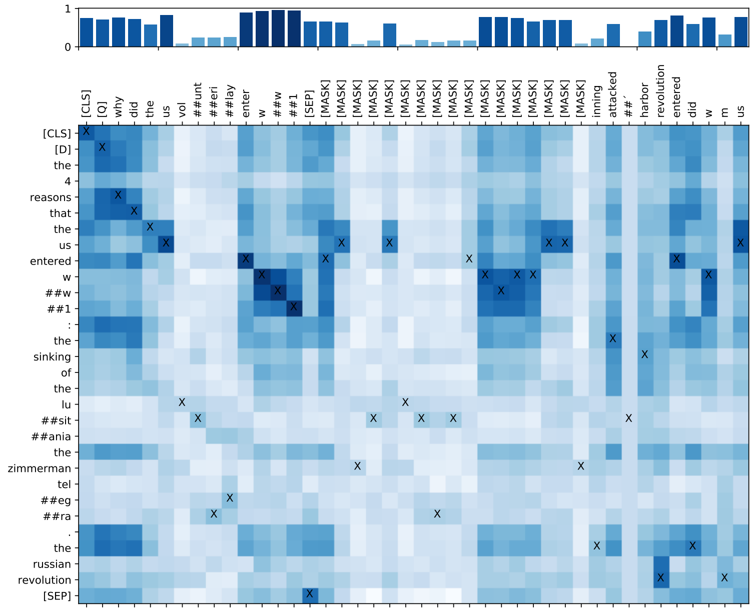


Fig. 6. ColBERT-PRF interaction matrix between query (qid: 106375) and passage (docid: 4337532) embeddings. The darker shading indicate a higher similarity. The highest similarity among all the passage embeddings for a given query embedding is highlight with a X symbol. The histogram depicts the magnitude of contribution for each query embedding to the final score of the passage.

Overall, in response to RQ3, 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 added to the query (f_b & f_e) as well as their relative importance during scoring (c.f. 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.

5.3.4 Results for RQ4 - 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[9] 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 used our default setting for the ColBERT-PRF model, i.e., selecting ten expansion embeddings; Equation (3) is used to resolve embeddings to tokens. On examination of Table 4, 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.

[9]In Table 4, 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.

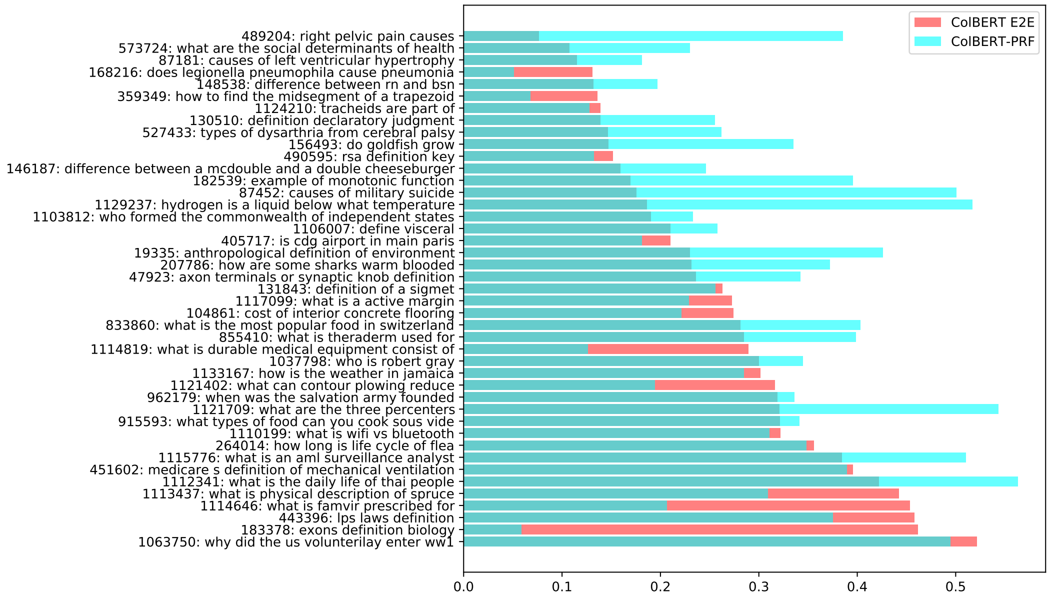
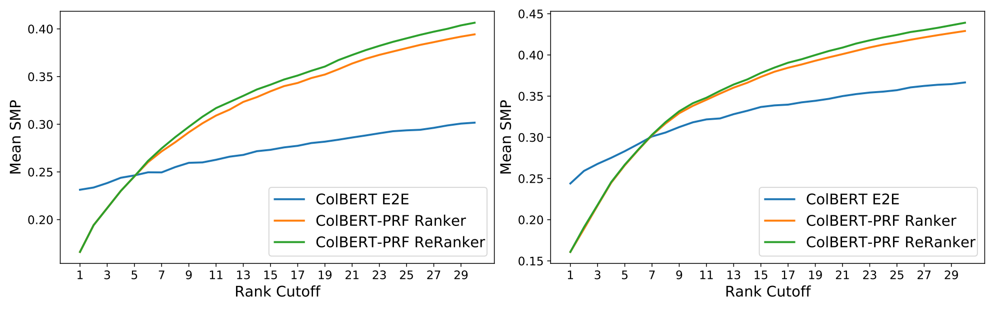


Fig. 7. Per-query semantic matching proportion measurements (measured to rank 10) for the ColBERT E2E (shown as red bars) and ColBERT-PRF (shown as cyan bars) models on the TREC 2019 passage ranking query set.



(a) Mean SMP on TREC 2019 (b) Mean SMP on TREC 2020

Fig. 8. Mean Semantic Matching Proportion (Mean SMP) as rank varies.

To answer RQ4, we further conduct analysis to measure the ability to perform semantic matching within the ColBERT Max-Sim operation. 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 [13], the authors concluded that ColBERT is able to conduct exact matches for important terms based on their embedded representations. In contrast, little work has considered the extent that ColBERT-based models perform semantic (i.e. non-exact) matching. Thus, firstly, following [28], we look into the interaction matrix between the query and passage embeddings. Figure 6 describes the interaction matrix between the query "why did the us voluntarily enter ww1 " expanded with 10 expansion embeddings and its top returned

Table 4. 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)

passage embeddings[10]. From Figure 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 Max-Sim scored passage tokens. In contract, the remaining query tokens are performing semantic matching to the passage as their corresponding passage tokens with the highest Max-Sim 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 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.

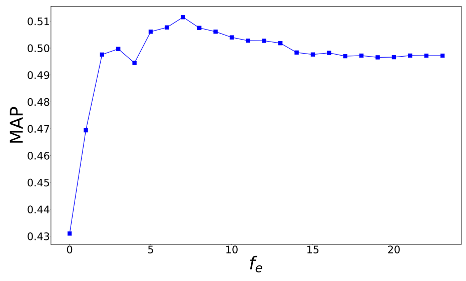
To quantify the extent that semantic matching takes place, we follow [43] and employ a recent measure that inspects the Max-Sim, and determines whether each query embedding is matched with the same token (exact match) vs. an inexact (semantic) match with a different token. Formally, let t_i and t_j respectively 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:

$$SMP(q, R_k) = \sum_{d \in R_k} \frac{\sum_{i \in \text{toks}(q)} \mathbb{1}[t_i \neq t_j] \cdot \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}}{\sum_{i \in \text{toks}(q)} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}}, \tag{6}$$

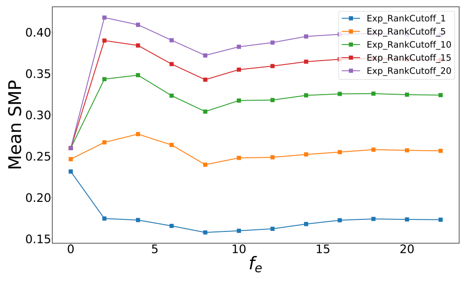
[10]We use a FAISS index to map embeddings back to most likely token.

where $\text{toks}(q)$ returns the indices of the query embeddings that correspond to BERT tokens, i.e., not [CLS], [Q], or [MASK] tokens[11], R_k is the top ranked k passages, and $\mathbb{1}[]$ is the indicator function.

Figure 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 8. In general, from Figure 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 decreased by rank (resulting in increasing SMP). However, that ColBERT-PRF (both Ranker and Reranker) have, in general, higher SMP than the original ColBERT ranking. This verifies the results from Figure 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 higher preference for exact token matches than the E2E baseline. However, overall, the higher SMPs exhibited by ColBERT-PRF indicates that, at deeper ranks, the embedding-based query expansion has the ability to retrieve passages with less lexical exact match between the query and passage embeddings.



(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.

Fig. 9. Potential topic drift analysis for ColBERT-PRF ReRanker on the TREC 2019 query set.

In addition, we further investigate the potential for topic drift when applying ColBERT-PRF with different number of expansion embeddings on the TREC 2019 queries. In particular, in Figure 9a[12] we measure retrieval effectiveness (MAP) as the number of expansion embeddings is varied and, in Figure 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 9a, we can see that $f_e = 8$ gives the highest (MAP) effectiveness (as also shown earlier in Figure 5b). At the same time, from Figure 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 are selected, excessive semantic matching occurs (Figure 9b) and effectiveness approaches MAP 0.50 (Figure 9a). As expansion embeddings are selected by using the IDF of the corresponding token,

[11]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.
[12]This is a subset of the curves presented earlier in Figure 5b, repeated here for ease of reference.

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.

6 DOCUMENT RANKING EFFECTIVENESS OF COLBERT-PRF

After assessing the effectiveness of ColBERT-PRF on passage ranking 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 chunck, with length comparable to those of passages. Moreover, in document 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 5.2, e.g., in a *zero shot* setting. Thus, in this section, we focus on testing the effectiveness of our proposed ColBERT-PRF for MSMARCO document retrieval task and the TREC Robust04 document retrieval task. Research questions and experimental setup for document ranking experiments are detailed in Section 6.1 and Section 6.2, respectively. Results and analysis are discussed in Section 6.3.

6.1 Research Questions

Our document ranking experiments address the following research questions:

- RQ5: Can our pseudo-relevance feedback mechanism enhance over 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?
- RQ6: 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?

6.2 Experimental Setup

6.2.1 Dataset & Measures. In this section, we evaluate our ColBERT-PRF on document ranking task using MSMARCO document and Robust04 document datasets. The MSMARCO training dataset contains ~3.2M documents along with 367K training queries, each with 1-2 labelled relevant documents. The Robust04 collection contains 528K newswire articles from TREC disks 4 & 5. To test retrieval effectiveness of 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 with an average of 153.4 and 39.26 relevant documents per query, respectively. In addition, we also conduct the evaluation using 250 title-only and description-only query sets from 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 5.2. For significance testing, we use the paired t-test ($p < 0.05$) and apply the Holm-Bonferroni multiple testing correction.

6.2.2 Implementation and Settings. As the length of documents in these corpora are too long to be fitted into the BERT [11] model, and in particular our trained ColBERT model[13] (limited to 512 and 180 BERT WordPiece tokens, respectively), we split long documents into smaller passages and index the generated passages following [9]. 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., its *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[14]. 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 5.2. For query expansion settings, we follow the default settings for passage ranking task in Section 5.3, which is 10 expansion terms obtained from 3 feedback passages[15] and $K = 24$ clusters.

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

6.3 Document Ranking Results

Table 5. Results for 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}	$\mathbf{0.4885}^{adghklm}$	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}

[13]It is a common practice to use models trained on the MSMARCO passage corpus [30] for document retrieval (e.g. [21, 31]).
[14]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 https://github.com/castorini/anserini/blob/master/src/main/python/passage_retrieval/example/robust04.md.
[15]We also tried filtering passages from the same document before applying PRF. We observed no significant improvements across multiple measures.

Table 6. Results for 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}

In this section, we further investigate the effectiveness of our proposed ColBERT-PRF for document ranking task. Table 5 and Table 6 present the performance of ColBERT-PRF models as well as the baselines on the MSMARCO document dataset and the Robust04 dataset, respectively.

6.3.1 Results for RQ5. 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 5, 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 [7].[16] In terms of NDCG@10, ColBERT-PRF outperforms over 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 6, 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, 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

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

the passage ranking task (Section 5). Thus, in response to RQ5, we conclude that our ColBERT-PRF is effective at improving ColBERT E2E on document ranking tasks, similar to the improvements observed in Section 5.

6.3.2 Results for RQ6. In the following, we compare the effectiveness of the ColBERT-PRF model with various baselines. From Table 5, 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 6, we observe that ColBERT-PRF models significantly outperforms the BM25 on both query sets and markedly outperforms over BM25 + RM3 on title-only queries. In addition, ColBERT-PRF show the similar performance with CEQE models in terms of MAP but exhibit markedly 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 exhibits comparable performance than 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 RQ6, we find that ColBERT-PRF is more effective than most of the baseline models and comparable to the BERT based neural reranking models.

7 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 7.1. Accordingly, a research question is posed in Section 7.2, with a corresponding experimental setup. Finally, Section 7.3 presents the performance and analysis of the three ColBERT-PRF variants.

7.1 Methodology

In Section 4.2 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 v_i generated by ColBERT-PRF. This results in a weight, $\sigma(v_i)$, that is used in the expanded max-sim calculation (Equation (5)). Indeed, notions of document frequency or collection frequency are commonly used in PRF models to measure expansion terms [2]. 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 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:

$$\sigma_{ICTF}(t) = \log\left(\frac{|D| + 1}{tf(t, D) + 1}\right) \tag{7}$$

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. 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:

$$\sigma_{MCos}(t) = \frac{1}{tf(t, D)} \sum_{j=1}^{tf(t,D)} \cos(\Upsilon, \phi_{c_j}) \tag{8}$$

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.

7.2 Research Question & Experimental Setup

Our informativeness measurement experiments address the following research question:

- RQ7: 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 RQ7, 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.

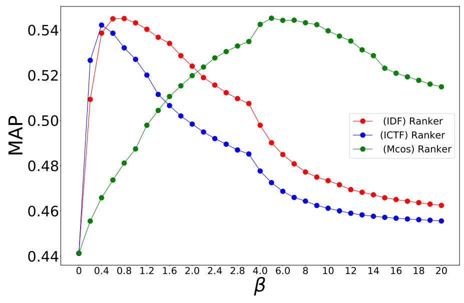
Dataset: The query sets we used to demonstrate the effectiveness of the three variants of ColBERT-PRF proposed are the MSMARCO passage TREC 2019 and TREC 2020 passage query sets for passage retrieval task and the Robust title and description query sets for document retrieval task.

Measures: Mean Average Precision (MAP) is used as the main metric.

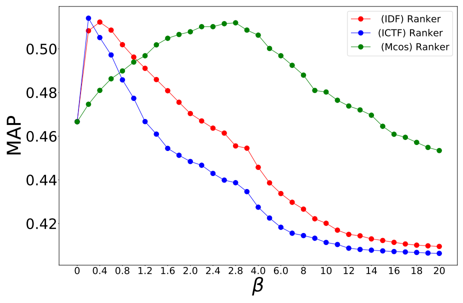
7.3 Results

Figure 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.

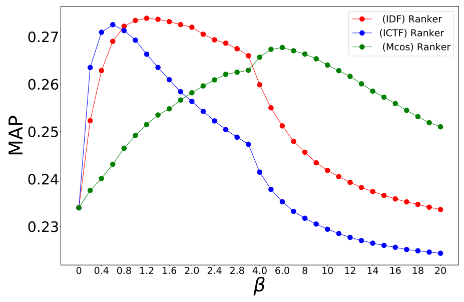
On analysing the figure, we see that, for both TREC 2019 and TREC 2020 query sets, the peak MAP scores for all the three weighting methods are the same, approximately with MAP=0.54 and MAP=0.51 respectively. In addition, according to the Figure 10a and 10b, the overall trend for IDF and



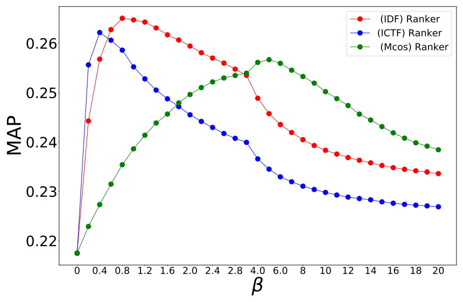
(a) MSMARCO passage TREC 2019 query set.



(b) MSMARCO passage TREC 2020 query set



(c) Robust04 title query set.



(d) Robust04 desciption query set.

Fig. 10. Influence of different weighting methods. $\beta = 0$ corresponds to the original ColBERT.

ICTF weighting methods are the same and both reaches 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 ~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 queryset (Figure 10c and 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 RQ7, we find that the statistical information, based 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.

8 EFFICIENT VARIANTS OF COLBERT-PRF

In Section 5.3, 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 8.1. Two research questions and the implementation setup are detailed in Section 8.2. Results and analysis are discussed in Section 8.3.

8.1 ColBERT-PRF variants

The overall workflow of a ColBERT-PRF Ranker model can be described into five stages, as shown in Figure 1. These stages can be summarised as follows (for the ColBERT-PRF ReRanker model, the fourth stage ANN retrieval is omitted):

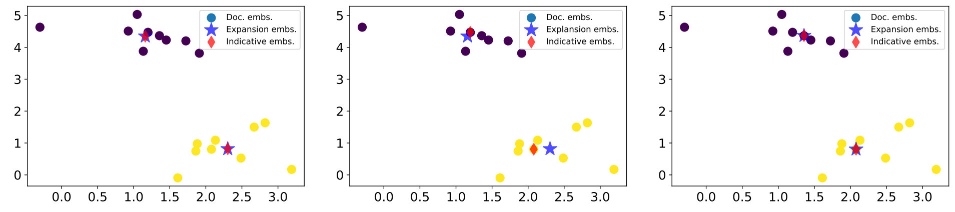
- Stage 1: First-pass FAISS ANN Retrieval
- Stage 2: First-pass exact ColBERT MaxSim re-ranking
- Stage 3: Clustering of Feedback Documents and Expansion Embedding Weighting
- Stage 4: Second-pass FAISS ANN Retrieval
- Stage 5: Second-pass exact ColBERT MaxSim re-ranking

In the following, we discuss changes to the clustering (Stage 3 above, Section 8.1.1) and ANN retrieval (Stages 1 & 4, Section 8.1.2).

8.1.1 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 11a provides an illustration of the KMeans clustering method. Indeed, as shown in Figure 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 require 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 (3) to obtain a list of token ids (see Section 4.2).

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



(a) KMeans clustering. (b) KMeans-Closest clustering. (c) KMedoid clustering.

Fig. 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.

use its token id to measure the importance score, such as IDF of the expansion embeddings. As shown in Figure 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 [18]. 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 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.

8.1.2 ANN Retrieval. The overall ColBERT-PRF Ranker process encapsulates a total of 5 stages, as shown in Figure 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 3, 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 (c.f. the contribution histogram in Figure 6)[17]. 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* [26] 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), its has has marked overall efficiency benefits [26] for ColBERT dense retrieval, as smaller number of candidates can be passed to MaxSim without loss of recall.

[17]Indeed, in separate but orthogonal work [41], 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.

More specifically, for the ColBERT-PRF instantiated as Ranker model, we apply the Approximate Scoring technique only in the first stage or in both the first and fourth stage 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 technique and the Approximate Scoring technique in the various ColBERT stages.

8.2 Research Question & Experimental Setup

- RQ8: 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?
- RQ9: What is the impact on efficiency and effectiveness of the ColBERT-PRF model when instantiated using Approximate Scoring?

Dataset: We compare the efficiency and the effectiveness of ColBERT-PRF model efficient variants on TREC 2019 and TREC 2020 query sets from MSMARCO passage.
Measures: 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 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 5.2, 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.
Experimental setup: 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 5, 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 $k_4 = 1000$[18], 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.

8.3 Results

8.3.1 RQ8 - Clustering Variants. Table 7 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 7, 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

[18]Indeed, [26] suggest $k = 300$ is sufficient for high precision retrieval.

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 7 (e.g. on TREC 2019, ~900ms for KMeans-Closest vs. ~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 RQ8, 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.

8.3.2 RQ9 - Variants using Approximate Scoring. Next, we consider the application of Approximate Scoring within ColBERT-PRF. Again, efficiency and effectiveness results are reported in Table 7. 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 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 outperform the ColBERT E2E model (but not by a significant margin). From Table 7, 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 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 ($100 \le \{k_1, k_3\} \le 7300$[19]). 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 the Stage 1 Approximate Scoring is applied. From Figure 12b, we observe that, for the smaller k_1, there is some minor degradation of NDCG@10; but the impact on MAP (Figure 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 12c and Figure 12d, where we applied the Approximate Scoring technique for both the first and fourth stages for ColBERT-PRF Ranker model, with the different clustering

[19]7300 is the average number of passages retrieved by ColBERT E2E for $k' = 1000$

Table 7. 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

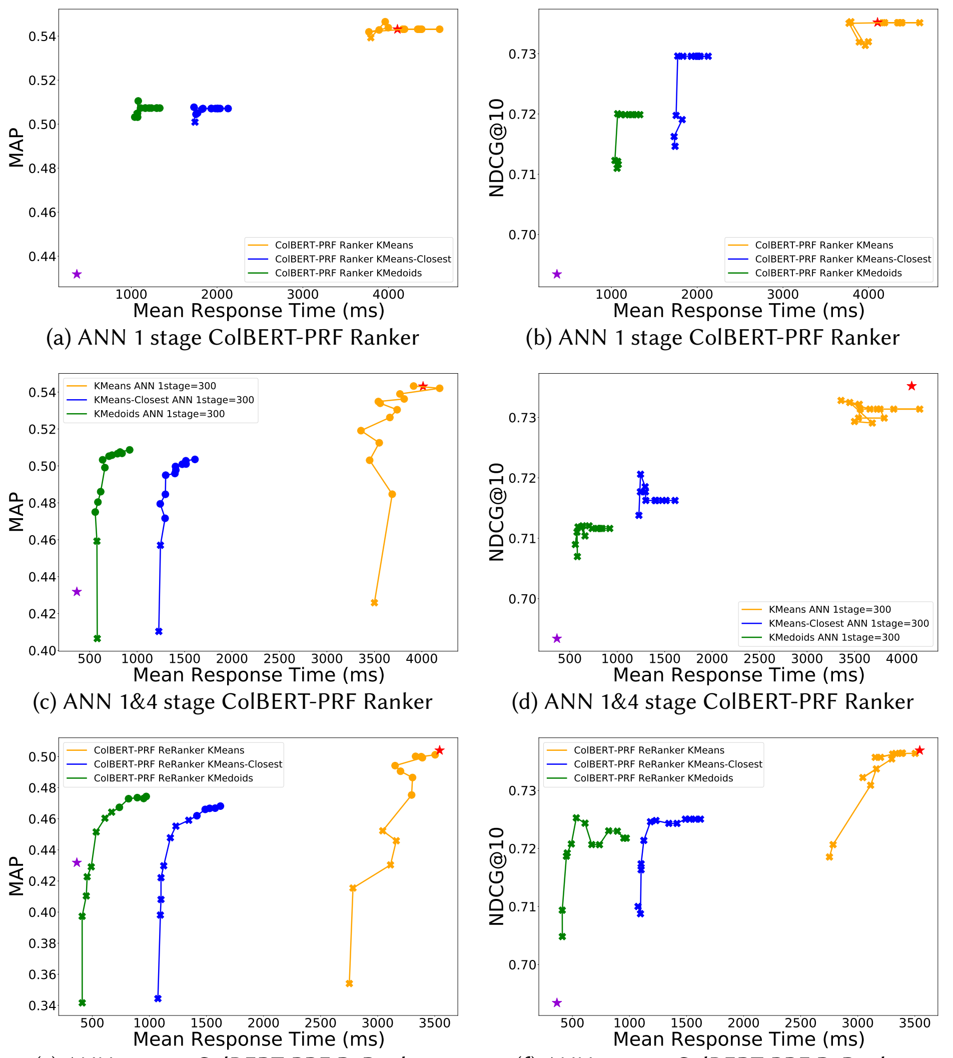
methods. More specifically, k_1, the rank cutoff of first stage Approximate Scoring is fixed to 300, while k_4 is varied. From Figure 12c, we find that all of the three clustering techniques exhibit correlations between efficiency and effectiveness, in that increased MRT also exhibits increased effectiveness. 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 12e & 12f, we observe that the trade-off curves for ColBERT-PRF ReRanker model with different clustering technique exhibits similar trend with Figure 12c & 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 RQ9, we conclude that the Approximate Scoring technique is useful to attain a better balance of effectiveness and efficiency for ColBERT-PRF model, by reducing the number of documents being re-ranked, and can also be combined with the more efficient clustering techniques.

9 CONCLUSIONS

This work is the first to propose 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



(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

Fig. 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.

beyond that of ColBERT. Indeed, our passage ranking experimental results – on the TREC 2019 and 2020 Deep Learning track passage ranking query sets – show 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. 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 validate 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 investigate 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 explore 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.

In conclusion, the main findings of this work 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 (Section 5) and document retrieval (Section 6). 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 7);
- 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 8).

Overall, our work 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, how this proposed dense PRF technique can be applied to the single-representation dense retrieval models remains an open problem. In addition, while the performance of most of the queries can benefit from the expansion embeddings, the performance of some of the queries is still degraded. Thus, a more cautious design that applies selective query embedding expansion will likely alleviate this issue. We leave this as one of our future works.

ACKNOWLEDGEMENTS

Nicola Tonellotto was partially supported by the Italian government in the framework of the Progetto PNRR “CN1 - Simulazioni, calcolo e analisi dei dati ad alte prestazioni - Spoke 1 - Future HPC & Big Data”. Xiao Wang acknowledges support by the China Scholarship Council (CSC) from the Ministry of Education of P.R. China. Craig Macdonald and Iadh Ounis acknowledge EPSRC grant EP/R018634/1: Closed-Loop Data Science for Complex, Computationally- & Data-Intensive Analytics.

The authors are thankful to reviewers for their constructive suggestions and comments, as well as Sean MacAvaney and Sasha Petrov for insightful comments and assistance with implementations.

REFERENCES

[1] Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *Proceedings of TREC*.

[2] Giambattista Amati. 2003. Probability models for information retrieval based on divergence from randomness Ph.D. thesis. *University of Glasgow* (2003).

[3] Giambattista Amati, Claudio Carpineto, and Giovanni Romano. 2004. Query difficulty, robustness, and selective application of query expansion. In *Proceedings of ECIR*. 127–137.

[4] Gianni Amati and Cornelis Joost Van Rijsbergen. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 357–389.
[5] David Arthur and Sergei Vassilvitskii. 2007. K-Means++: The Advantages of Careful Seeding. In *Proceedings of SODA*. 1027–1035.
[6] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. 2008. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of SIGIR*. 243–250.
[7] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the TREC 2020 deep learning track. In *Proceedings of TREC*.
[8] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the TREC 2019 deep learning track. In *Proceedings of TREC*.
[9] Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for IR with contextual neural language modeling. In *Proceedings of SIGIR*. 985–988.
[10] Zhuyun Dai and Jamie Callan. 2020. Context-aware document term weighting for ad-hoc search. In *Proceedings of WWW*. 1897–1907.
[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of ACL*. 4171–4186.
[12] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. 2016. Query Expansion with Locally-Trained Word Embeddings. In *Proceedings of ACL*. 367–377.
[13] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. A White Box Analysis of ColBERT. In *Proceedings of ECIR*. 257–263.
[14] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of CIKM*. 55–64.
[15] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).
[16] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of EMNLP*. 6769–6781.
[17] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of SIGIR*. 39–48.
[18] Ilyes Khennak, Habiba Drias, Amine Kechid, and Hadjer Moulai. 2019. Clustering algorithms for query expansion based information retrieval. In *Proceedings of ICCI*. 261–272.
[19] Saar Kuzi, Anna Shtok, and Oren Kurland. 2016. Query expansion using word embeddings. In *Proceedings of CIKM*. 1929–1932.
[20] Canjia Li, Yingfei Sun, Ben He, Le Wang, Kai Hui, Andrew Yates, Le Sun, and Jungang Xu. 2018. NPRF: A Neural Pseudo Relevance Feedback Framework for Ad-hoc Information Retrieval. In *Proceedings of EMNLP*. 4482–4491.
[21] Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2021. PARADE: Passage Representation Aggregation for Document Reranking. arXiv:2008.09093 [cs.IR]
[22] Hang Li, Shengyao Zhuang, Ahmed Mourad, Xueguang Ma, Jimmy Lin, and Guido Zuccon. 2021. Improving Query Representations for Dense Retrieval with Pseudo Relevance Feedback: A Reproducibility Study. In *Proceedings of ECIR*.
[23] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics* 9 (2021), 329–345.
[24] Sean MacAvaney. 2020. OpenNIR: A Complete Neural Ad-Hoc Ranking Pipeline. In *Proceedings of WSDM*. 845–848.
[25] Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation in Information Retrieval Using PyTerrier. In *Proceedings of ICTIR*. 161–168.
[26] Craig Macdonald and Nicola Tonellotto. 2021. On approximate nearest neighbour selection for multi-stage dense retrieval. In *Proceedings of CIKM*. 3318–3322.
[27] Craig Macdonald, Nicola Tonellotto, Sean MacAvaney, and Iadh Ounis. 2021. PyTerrier: Declarative Experimentation in Python from BM25 to Dense Retrieval. In *Proceedings of CIKM*. 4526–4533.
[28] Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2021. On Single and Multiple Representations in Dense Passage Retrieval. *IIR 2021 Workshop* (2021).
[29] Shahrzad Naseri, Jeffrey Dalton, Andrew Yates, and James Allan. 2021. CEQE: Contextualized Embeddings for Query Expansion. *Proceedings of ECIR* (2021), 467–482.
[30] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPs*.
[31] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713* (2020).
[32] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[33] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375* (2019).

[34] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Douglas Johnson. 2005. Terrier information retrieval platform. In *Proceedings of ECIR.* 517–519.

[35] Ramith Padaki, Zhuyun Dai, and Jamie Callan. 2020. Rethinking query expansion for BERT reranking. In *Proceedings of ECIR.* 297–304.

[36] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of NAACL-HLT.* 2227–2237.

[37] Joseph Rocchio. 1971. Relevance feedback in information retrieval. *The Smart Retrieval System-experiments in Automatic Document Processing* (1971), 313–323.

[38] Dwaipayan Roy, Sumit Bhatia, and Mandar Mitra. 2019. Selecting Discriminative Terms for Relevance Model. In *Proceedings of SIGIR.* 1253–1256.

[39] Dwaipayan Roy, Debasis Ganguly, Sumit Bhatia, Srikanta Bedathur, and Mandar Mitra. 2018. Using word embeddings for information retrieval: How collection and term normalization choices affect performance. In *Proceedings of CIKM.* 1835–1838.

[40] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. 2016. Using word embeddings for automatic query expansion. In *Proceedings of SIGIR Workshop on Neural Information Retrieval.* arXiv:1606.07608.

[41] Nicola Tonellotto and Craig Macdonald. 2021. Query Embedding Pruning for Dense Retrieval. In *Proceedings of CIKM.* 3453–3457.

[42] Junmei Wang, Min Pan, Tingting He, Xiang Huang, Xueyan Wang, and Xinhui Tu. 2020. A Pseudo-relevance feedback framework combining relevance matching and semantic matching for information retrieval. *Information Processing & Management* 57, 6 (2020), 102342.

[43] Xiao Wang, Craig Macdonald, and Iadh Ounis. 2022. Improving zero-shot retrieval using dense external expansion. *Information Processing & Management* 59, 5 (2022), 103026.

[44] Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2021. Pseudo-Relevance Feedback for Multiple Representation Dense Retrieval. In *Proceedings of ICTIR.* 297–306.

[45] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of SIGIR.* 55–64.

[46] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of ICLR.*

[47] HongChien Yu, Zhuyun Dai, and Jamie Callan. 2021. PGT: Pseudo Relevance Feedback Using a Graph-Based Transformer. In *Proceedings of ECIR.* 440–447.

[48] HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. Improving Query Representations for Dense Retrieval with Pseudo Relevance Feedback. In *Proceedings of CIKM.* 3592–3596.

[49] Hamed Zamani and W Bruce Croft. 2016. Embedding-based query language models. In *Proceedings of ICTIR.* 147–156.

[50] Hamed Zamani, Mostafa Dehghani, W Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. In *Proceedings of CIKM.* 497–506.

[51] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: Contextualized Query Expansion for Document Re-ranking. In *Proceedings of EMNLP: Findings.* 4718–4728.

A APPENDIX

In the following, Appendix A.1 firstly details how variants of ColBERT-PRF can be implemented which weight token occurrences, specifically using the Bo1 and RM3 query expansion models. These are compared with ColBERT-PRF implemented with KMeans clustering technique. Next, in Appendix A.2, we demonstrate the experimental pipelines for ColBERT-PRF .

A.1 ColBERT-PRF (Bo1 or RM3) variants

As discussed in Section 4.1, in ColBERT-PRF, embeddings are clustered, rather than the frequency of the corresponding tokens. In this section, we analyse this choice, by separating the clustering from the embedded representation. In particular, we use traditional token counting to measure the informativeness of tokens in the feedback documents, but then expand the query using the corresponding embedded representation of the selected token. Therefore, for these variants, the informativeness of each feedback embedding is measured using the Bo1 or RM3 technique, then the highest informativeness feedback embeddings are selected as the expansion embeddings. For instance, for the ColBERT-PRF (Bo1) implementation, the expansion embeddings are weighted according to the following equation:

$$W_{\text{Bo1}}(t) = tf_x \log_2 \frac{1+\lambda}{\lambda} + \log_2(1+\lambda), \tag{9}$$

where $\lambda = tf_{rel}/N_{rel}$, tf_{rel} denotes the frequency of (BERT WordPiece) token t in the pseudo-relevant feedback documents and N_{rel} denotes the number of feedback documents. tf_x denotes the number of unique tokens in the pseudo-relevant document set.

Similarly, for the ColBERT-PRF (RM3) variant, the expansion embeddings are selected using:

$$W_{\text{RM3}}(t) = \lambda \text{ score }_{\text{exp}}(t) + (1-\lambda) \text{ score }_{\text{orig}} (t), \tag{10}$$

where $0 \leq \lambda \leq 1$. score $_{\text{orig}}$ $(t) = 1$ denotes the weights for the original query embeddings and $\text{score}_{\text{exp}}(t)$ denotes the weights for the expansion embeddings. $\text{score}_{\text{exp}}(t) = \frac{S(t)}{\sum_{d \in PRD} \sum_{t' \in d} S(t')}$, where $S(t)$ is calculated as follows:

$$\begin{aligned} S(t) &= P\left(t, q_1, \ldots, q_{|q|}\right) \\ &= \sum_{M \in \mathcal{M}} P(M) P(t \mid M) \prod_{i=1}^{|q|} P\left(q_i \mid M\right) \\ &= \frac{1}{\#PRD} \sum_{d \in PRD} \left(\frac{tf(t,d)}{|d|} \times MaxSim(q,d)\right) \end{aligned}, \tag{11}$$

where $MaxSim(q,d) = \sum_{i=1}^{|q|} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}$ denotes the maximum similarity score.

In Table 8, we report the ColBERT-PRF models with various expansion embedding selection techniques on both the TREC 2019 and 2020 query sets. From Table 8, we find that both ColBERT-PRF Ranker and ReRanker with the Bo1 selection technique can outperform the ColBERT E2E model in terms of NDCG@10, MRR@10 and Recall on TREC 2019 while the improvements are not observed on the TREC 2020 query set. In particular, on the TREC 2020 query set, the ColBERT-PRF models with the RM3 expansion embeddings selection approach exhibit lower performance than the ColBERT E2E model. More importantly, we observe that, by comparing the ColBERT-PRF model with the KMeans clustering technique with the Bo1 and RM3 selection variants, the KMeans clustering technique significantly outperforms both the Bo1 and RM3 variants. Figure 13 below shows the impact of β for the Bo1 and KMeans variants, in both the ranking and reranking settings, for MAP and NDCG@10. From the figures, it is clear that KMeans always outperforms Bo1, regardless of the setting of β.

Thus, we conclude that the KMeans clustering selection technique is more effective than the traditional Bo1 and RM3 selection approaches for the ColBERT-PRF model. This is because the Bo1 and RM3 query expansion techniques rely solely on the word occurrence statistics for selecting expansion embeddings rather than the semantic coherence of the embeddings, and hence select embeddings for tokens that occur frequently, rather than for frequently occurring semantic concepts. Selecting semantically coherent concepts is a key advantage of ColBERT-PRF for a dense retrieval environment.

A.2 ColBERT-PRF Pipeline

In this appendix, we demonstrate the stages of ColBERT-PRF, when defined as PyTerrier [25, 27] pipelines. In particular, in PyTerrier, the >> operator is used to delineate different stages of a retrieval

Table 8. Comparison of applying clustering-based expansion embedding selection vs. Bo1 and RM3 based expansion embedding selection for ColBERT-PRF. Superscripts a...e 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	MRR@10	Recall	MAP	NDCG@10	MRR@10	Recall
ColBERT E2E (a)	0.4318	0.6934	0.8529	0.7892	0.4654	0.6871	0.8525	0.8245
ColBERT-PRF Ranker (Bo1) (b)	0.4720	0.7068	0.8420	0.8561	0.4609	0.6811	0.8374	0.8589^{c}
ColBERT-PRF Ranker (RM3) (c)	0.4036	0.7352	0.8858	0.8706	0.4289	0.6483	0.8417	0.7852
ColBERT-PRF Ranker (KMeans)	$\mathbf{0.5427}^{abc}$	**0.7395**	**0.8897**	$\mathbf{0.8711}^{ac}$	$\mathbf{0.5116}^{c}$	**0.7153**	**0.8439**	$\mathbf{0.8837}^{abc}$
ColBERT-PRF ReRanker (Bo1) (d)	0.4382	0.7096	0.8411	0.7891	0.4650	0.6819	0.8374	0.8278^{c}
ColBERT-PRF ReRanker (RM3) (e)	0.3943	0.6686	0.8624	0.7281	0.4313	0.6502	0.8417	0.7882
ColBERT-PRF ReRanker (KMeans)	$\mathbf{0.5026}^{abc}$	**0.7409**	**0.8897**	$\mathbf{0.7977}^{c}$	**0.5063**	**0.7161**	**0.8439**	$\mathbf{0.8443}^{c}$

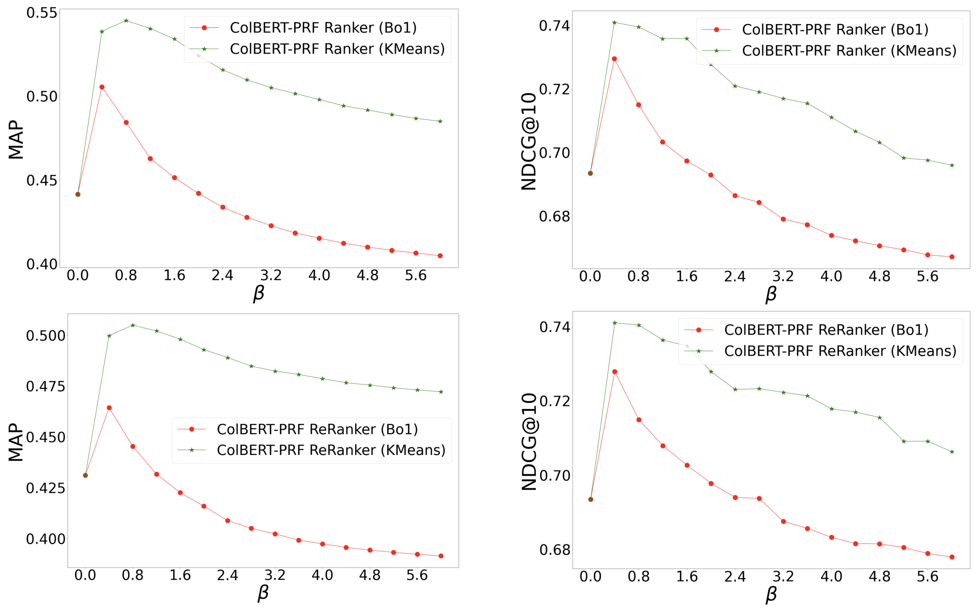


Fig. 13. Impact of β on TREC 2019 query set for ColBERT-PRF Ranker and ReRanker models with Bo1 and RM3 expansion embedding selection techniques in terms of MAP and NDCG@10 performances.

pipeline. 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.[20]

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,
```

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

```
                                    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
```