Reproducibility, Replicability, and Insights into Dense Multi-Representation Retrieval Models: from ColBERT to Col★

Xiao Wang
University of Glasgow, UK
x.wang.8@research.gla.ac.uk

Craig Macdonald
University of Glasgow, UK
craig.macdonald@glasgow.ac.uk

Nicola Tonellotto
University of Pisa, Italy
nicola.tonellotto@unipi.it

Iadh Ounis
University of Glasgow, UK
iadh.ounis@glasgow.ac.uk

ABSTRACT

Dense multi-representation retrieval models, exemplified as ColBERT, estimate the relevance between a query and a document based on the similarity of their contextualised token-level embeddings. Indeed, by using contextualised token embeddings, dense retrieval, conducted as either exact or semantic matches, can result in increased effectiveness for both in-domain and out-of-domain retrieval tasks, indicating that it is an important model to study. However, the exact role that these semantic matches play is not yet well investigated. For instance, although tokenisation is one of the crucial design choices for various pretrained language models, its impact on the matching behaviour has not been examined in detail. In this work, we inspect the reproducibility and replicability of the contextualised late interaction mechanism by extending ColBERT to Col★, which implements the late interaction mechanism across various pretrained models and different types of tokenisers. As different tokenisation methods can directly impact the matching behaviour within the late interaction mechanism, we study the nature of matches occurring in different Col★ models, and further quantify the contribution of lexical and semantic matching on retrieval effectiveness. Overall, our experiments successfully reproduce the performance of ColBERT on various query sets, and replicate the late interaction mechanism upon different pretrained models with different tokenisers. Moreover, our experimental results yield new insights, such as: (i) semantic matching behaviour varies across different tokenisers; (ii) more specifically, high-frequency tokens tend to perform semantic matching than other token families; (iii) late interaction mechanism benefits more from lexical matching than semantic matching; (iv) special tokens, such as `[CLS]`, play a very important role in late interaction.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**.

KEYWORDS

Dense Retrieval; Semantic Matching; Reproducibility

ACM Reference Format:
Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2023. Reproducibility, Replicability, and Insights into Dense Multi-Representation Retrieval Models: from ColBERT to Col★. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591916

1 INTRODUCTION

Ranking is a core task in information retrieval systems. Traditional lexical retrieval models, such as BM25, focus upon exact term matching, where the relevance of a document to a search query is measured by their precise term overlap. However, this causes both vocabulary and semantic mismatch problems during retrieval. On the one hand, there is a high chance that users would formulate their queries using different terms than used in the relevant document(s), as users have no access to the relevant documents prior to the search. For instance, a document describing a 'vehicle' may not be scored highly for a query 'car' – an example of the vocabulary mismatch problem. On the other hand, precise lexical matching cannot distinguish the same word with different senses. For instance, the word 'case' exhibits different meanings when used in phrases like "phone case" or "case study" phrases. Such a polysemous word can cause problems for retrieval.

Recently proposed dense retrieval models alleviate the above limitations by encoding the query and document into contextualised embeddings, and have yielded significant improvements over lexical retrieval [12, 13, 18, 38, 39]. In dense retrieval, the relevance of a document to a query is estimated according to the inner product of the corresponding contextualised embeddings in the same vector space. Most dense retrieval models encode queries and documents as single-representation embeddings, i.e., a single vector to represent a document or a query. Differently, ColBERT [13] encodes queries and documents into multiple representations, one vector per token. Then, ColBERT employs a *late interaction* scoring mechanism to estimate a similarity score between the query and document. ColBERT and its late interaction mechanism is an important dense retrieval paradigm as it shows high retrieval effectiveness on in-domain and zero-shot out-of-domain retrieval tasks, while also being flexible to perform other tasks, such as question answering and document retrieval [13, 17, 29]. Thus, in this paper, we take a closer look at ColBERT in terms of its "complete" [35] reproducibility (different team, same artefacts) and replicability (different team, different artefacts) [24] and evaluate its performance not only on the original paper used MSMARCO Dev query set but also on both

SIGIR '23, July 23–27, 2023, Taipei, Taiwan.

TREC DL 2019 & 2020 query sets. In addition, we conduct several ablation studies to further explain the performance of the model.

Moreover, the matching behaviour performed within the late interaction mechanism includes both lexical and semantic matching, both of which depend on the vocabulary. Sub-word tokenisation is the de-facto standard tokenisation approach in neural IR, due to the advantages of a limited-size vocabulary. Tokenisation algorithms used by common contextualised models include WordPiece [31], used by BERT [4] and ELECTRA [3], Byte-Pair Encoding (BPE) [32], used by RoBERTa [20], and SentencePiece [14], used by ALBERT [15] and T5 [27]. Different pretrained models, and their different tokenisation algorithms, lead to different embeddings in different representation spaces. In addition, the same type of pretrained model can often be instantiated in differing sizes (number of layers, etc.), where larger models can be more effective. Going further, we extend ColBERT to Col★, instantiating the late interaction mechanism with various pretrained models using different types of tokenisation techniques. By doing so, we further inspect the replicability [24] of the contextualised late interaction mechanism.

In practice, when scoring a query-document pair using the late interaction mechanism, some of the matched embeddings will represent the same token – a *lexical match* – while others may match at a *semantic* level ('car' vs. 'vehicle'). However, the extent that such *semantic matching behaviour* occurs among the contextualised representations is still under-investigated. Indeed, it is difficult to disentangle the semantic matching from the dot product operation on the single-representation dense retrieval models, while the multiple-representation dense retrieval paradigm provides more transparency in its ranking mechanism. Thus, in this work, we further inspect the dense matching behaviour in ColBERT as well as the Col★ models with different types of tokenisation methods, attempting to generate new insights behind the late interaction mechanism based retrieval. In particular, we investigate the matching behaviour for different token families and further quantify the contribution of different types of matching behaviour to the retrieval effectiveness.

In summary, this work studies the reproducibility and replicability of the ColBERT model. In addition, it provides new insights by explaining the semantic matching behaviour of the contextualised late interaction mechanism. Our main findings can be summarised in terms of *Reproducibility*, *Replicability* and *Insights* aspects of the contextualised late interaction mechanism, as follows:

Reproducibility: We investigate the reproducibility of ColBERT by training our own ColBERT models and: (i) we find that we are able to reproduce the results of ColBERT on MSMARCO Dev query set; (ii) in terms of the similarity function for ColBERT, we find that there is no difference between L2-based and Cosine similarity methods for reranking, but the L2 similarity method benefits more in end-to-end settings; (iii) regarding the number of training iterations, we find that ColBERT training becomes stable at around 150k iterations with a batch size of 32. However, further training beyond this point still results in a modest increase in retrieval effectiveness.

Replicability: We study the effectiveness of multi-representation dense retrieval with different pretrained models with different tokenisation algorithms and (iv) we find that ColBERT can generalise upon various pretrained language models. (v) in terms of retrieval effectiveness, we find that applying the late interaction mechanism upon a RoBERTa model (which employs BPE tokenisation) exhibits competitive retrieval effectiveness to ColBERT.

Insights: Extensive experimental analysis on semantic matching behaviour yields the following new findings: (vi) applying the late interaction mechanism with the BPE tokeniser is more likely to perform semantic matching than the more common ColBERT model; (vii) among various salient token families, all of the contextualised late interaction models perform semantic matching, particularly for low IDF tokens and stopwords tokens; (viii) performing only exact matching and the special token matching contribute more than only semantic matching to the overall retrieval effectiveness. These insights help explain the matching behaviour in contextualised late interaction retrieval and can shed light on the more effective dense retrieval model design and retrieval.

The remainder of this paper is organised as follows: Section 2 describes related work about dense retrieval and tokenisation. We detail the reproducibility and replicability experiments in Section 3 and Section 4, respectively. Next, we explain the semantic matching behaviour of the contextualised late interaction mechanism and generate new insights in Section 5. Finally, we summarise our findings and provide future work directions in Section 6.

2 RELATED WORK

Dense retrieval performs relevance scoring through the encoded contextualised representations of queries and documents. According to the way the queries and the documents are encoded, dense retrieval models can be divided into two families [22]: single representation and multiple representation dense retrieval models. In single representation models, such as DPR [12], ANCE [38] and TCT-ColBERT [19], each query or document as a whole is encoded into a single dense representation. Then the relevance between the query and document is estimated using the dot-product of the encoded vectors. In contrast, in multiple representation dense retrieval models, exemplified by ColBERT [13], each token of the query or document is encoded into a dense representation. To estimate the relevance score of a document to a query, ColBERT implements a two-stage scoring pipeline: in the first stage, an approximate nearest neighbour search produces a set of candidate documents, and in the second stage, these documents are re-ranked with a late interaction mechanism. Recent research has focused on various aspects to improve the quality of dense retrieval models. For instance, some researchers have studied the effect of the negative samples for training more effective dense retrieval models [19, 25, 38, 39]; other researchers have observed that distilling the knowledge from a more effective model, for instance, ColBERT, can result in a more effective single-representation dense retrieval model [9, 10, 18, 19, 35]. Another thread of work involves reducing the index size of ColBERT, for instance by pruning [16] or compressing embeddings [1, 29]. Most relevant to this work, Formal et al. investigated the term importance captured by ColBERT in the exact and semantic matches [5]. However, they did not investigate the importance of semantic matching, nor the impact of different base models and tokenisation methods. Therefore, our replicability study is important as it examines the generalisation of ColBERT to other pretrained language models and tokenisers. Later, Formal et al. attempted to quantify the importance of lexical matching by examining the frequency of important query tokens in the top-k returned documents [6]. However, it is unclear how strongly this metric correlates with the actual importance of lexical matching. In this work, we propose a semantic

Table 1: Tokenisation for example inputs for 3 tokenisers, corresponding to BERT, ALBERT and RoBERTa respectively.

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

matching proportion method that directly measures the extent a query token performs exact or semantic matching to the document.

Indeed, tokenisation is an important technique to preprocess the input text before input to a contextualised language model. In particular, as transformer-based models learn representations for each unique token, a limited-size vocabulary is important. A large vocabulary size would cause increased memory and time complexity, and difficulty of learning accurate representations for rare tokens. For these reasons, sub-word tokenisation is usually used to split the input text into small chunks of text. Thus, frequently-used words are given unique ids, while rare words will be processed into sub-words. Prevalent tokenisation techniques used by large pretrained language models include WordPiece [31], Byte-Pair Encoding (BPE) [32] and SentencePiece [14] tokenisation techniques. For instance, WordPiece [31] is used by BERT [4] and miniLM [34]; BPE [32] is used by RoBERTa [20] and GPT [26] models; SentencePiece [14] is used by ALBERT [15] and T5 [27] models. In particular, the BPE [32] and WordPiece [31] tokenisation technique merge the characters into larger tokens but control the vocabulary size using different algorithms to maximise the likelihood of the training data. In contrast, SentencePiece treats the whole sentence as one large token and learns to split it into sub-words.

Table 1 compares the outputs of the different tokenisation approaches for the example texts "`casualties in ww2`" and "`Casualties`". Firstly, each tokeniser has its own rule to mark the begin and end of the sentence and whether the token is sub-word token or not (## vs. _ vs. Ġ). Moreover, we see that all three compared tokenisation techniques can produce tokens of the more frequent words with their surface word form, such as `in`. However, for the rarer words (`ww2`), the various tokenisers differ in how they split these words into sub-words and encode as tokens. For instance, WordPiece and BPE produce separate the `w`, `w` and `2` in `ww2`, while SentencePiece has a token for `ww`. Notably, RoBERTa's BPE tokeniser is case-sensitive (see also Table 3), and while the vocabulary contains the surface form of `casulaties`, the uppercase word is broken into three sub-word tokens. This can directly impact the matching behaviour within the late interaction mechanism, as further discussed in Section 5.1.

Indeed, different tokenisers will directly affect the generated embeddings thus affecting the model performance. For instance, studies have examined different tokenisation techniques for language model pretraining [2, 8] and for low-resource language models [28, 33]. However, the impact of differing tokenisers for dense retrieval has not been previously investigated. Most recently, ColBERT-X [23] has replaced the BERT pretrained model with the XLM-RoBERTa pretrained model of ColBERT for the cross-language retrieval task. However, ColBERT-X is motivated by the cross-language abilities of the XLM-RoBERTa model and made no conclusions on the effect of the different tokenisation techniques. In this work, we not only investigate the effect of the different pretrained models in ColBERT but also study the effect of using different tokenisation techniques upon English dense retrieval. In addition, we further inspect their impact on the contextualised matching pattern occurring in the dense retrieval models.

3 REPRODUCIBILITY OF COLBERT

In this section, we first illustrate the late interaction mechanism implemented by ColBERT in Section 3.1, then detail the reproduction results from Section 3.2 to Section 3.4. In particular, the reproduction results address the following research questions: **RQ1.1:** Can we reproduce the training of ColBERT? (Section 3.2) Going further, we conduct ablations of ColBERT, including: **RQ1.2:** What is the impact of the similarity function for ColBERT? (Section 3.3) and **RQ1.3** Does the model really need to train with full 200k iterations (with batch size set as 32)? (Section 3.4). The source code, runs and model checkpoints for all of our experiments are provided in our virtual appendix.[1]

3.1 Contextualised Late Interaction

ColBERT consists of a query encoder E_Q and a document encoder E_D, which are fined-tuned based on the pretrained BERT model. For each query q and document d, the WordPiece tokeniser splits the query text into $\{t_{q_1}, t_{q_2}, \cdots t_{q_{|q|}}\}$ tokens and the document text into $\{t_{d_1}, t_{d_2}, \cdots t_{d_{|d|}}\}$ tokens. Then, the series of query and document tokens are encoded by the corresponding encoder into a bag of dense representations $\{\phi_{q_1}, \ldots, \phi_{q_{|q|}}\}$ and $\{\phi_{d_1}, \ldots, \phi_{d_{|d|}}\}$, respectively. In particular, the number of encoded query tokens is fixed to $|q| = 32$ and filled with the special token '`[MASK]`' if the original query contains less than 32 tokens. Moreover, a linear layer is used to map the BERT representations into a low-dimensional vector with m components, typically $m = 128$ [13].

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

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

where $Sim(.,.)$ denotes the similarity function used to measure the similarity between query and document embeddings. There are several commonly used similarity functions used for dense retrieval models, namely the L2-based and Cosine similarity[2] functions.

3.2 RQ1.1: Reproduce the Training of ColBERT

The aim of this section is to study the reproducibility of the BERT-based late interaction model, in particular, the ColBERT-v1 [13] model. We note that ColBERT-v2 [29] also uses the same late interaction mechanism of ColBERT-v1 while boosting its retrieval effectiveness by leveraging a number of tricks during training, including periodically mining hard-negative samples from the ColBERT-v2 indices [38], in-batch negative training and performing knowledge distillation from a MiniLM [34] based cross-encoder model. As the efficacy of the above training tricks has been studied in [18, 19, 25, 35, 38, 39], we focus upon the contextualised late interaction mechanism.

[1] https://github.com/Xiao0728/ColStar_VirtualAppendix

[2] This is implemented using the inner product, as the embeddings have been normalised to unit length.

Table 2: Reproduction results of ColBERT reranking (rerank on the official top-1000 results produced by BM25) and end-to-end retrieval on MSMARCO Dev Small as well as the TREC DL 2019 and 2020 query sets. The † symbol denotes statistically significant differences over BM25. The highest value in each column is boldfaced.

Models	MSMARCO (Dev Small)				TREC DL 19				TREC DL 20			
	MRR@10	R@50	R@100	R@1k	nDCG@10	MRR@10	MAP@1k	R@1k	nDCG@10	MRR@10	MAP@1k	R@1k
	BM25 (official) » Late Interaction				BM25 (PyTerrier) » Late Interaction				BM25 (PyTerrier) » Late Interaction			
BM25 (official)	0.167	-	-	0.814	-	-	-	-	-	-	-	-
BM25 (PyTerrier)	0.196	0.604	0.755	0.871	0.480	0.640	0.286	0.755	0.494	0.615	0.293	0.807
ColBERT-L2 (reported)	0.348	0.753	**0.805**	0.814	-	-	-	-	-	-	-	-
ColBERT-Cosine (reported)	0.349	-	-	-	-	-	-	-	-	-	-	-
ColBERT-L2 (ours)	**0.349**	**0.754**	**0.805**	0.814	**0.713**†	**0.862**†	**0.470**†	0.755	0.698†	0.828†	0.483†	0.807
ColBERT-Cosine (ours)	0.348	0.753	0.804	0.814	**0.713**†	0.847†	0.459†	0.755	**0.707**†	**0.835**†	**0.484**†	0.807
	ANN Search » Late Interaction											
ColBERT-L2 (reported)	**0.367**	0.829	0.923	**0.968**	-	-	-	-	-	-	-	-
ColBERT-L2 (ours)	0.361	**0.832**	**0.923**	0.965	**0.722**†	**0.870**†	**0.462**†	**0.823**	0.685†	0.823†	**0.475**†	**0.839**
ColBERT-Cosine (ours)	0.358	0.823	0.911	0.952	0.708†	0.857†	0.445†	0.773	**0.690**†	**0.832**†	0.473†	0.806

As the original authors did not release the trained query or document vectors nor a trained ColBERT model, thus we skip the "Last-Meter" and "Last-Mile" [35] reproduction stages. Building upon the original ColBERT paper [13], we conduct a "complete" reproduction [35] of ColBERT by training from scratch using the MSMARCO passage training dataset.[3] The MSMARCO training dataset contains ~8.8M passages along with 0.5M training queries, each with 1-2 labelled relevant passages. We train ColBERT using both similarity methods for 200k iterations with batch size set as 32. Following ColBERT, the retrieval effectiveness is measured on the MSMARCO Dev small query set, which contains 6980 queries with an average of 1.1 relevance judgements per query. We report MRR@10 and Recall with various rank cutoff values (R@50, R@200, R@1k) for MSMARCO Dev Small query set.

Results of RQ1.1: Table 2 reports the reproducibility results by training ColBERT using both Cosine and L2 similarity on MSMARCO Dev Small query set. From the table, we see that for the reranking setting, where ColBERT is applied to rerank the official top 1000 BM25 results, our trained ColBERT-L2 model exhibits negligible differences in terms of all the reported metrics (with a difference up to 0.012 on MRR@10) compared to the reported ColBERT-L2 model. Similarly, results for our trained ColBERT-Cosine model show that we can reproduce the original training of ColBERT for reranking. For the end-to-end setting, our trained ColBERT-L2 model can reproduce the results reported in the original paper.

Answer to RQ1.1: In summary, we find that we are able to reproduce the results of ColBERT on the MSMARCO Dev Small query set for both reranking and end-to-end scenarios by training our own ColBERT models.

3.3 RQ1.2: ColBERT Similarity Functions

For the choice of the vector-similarity method used by the MaxSim operator in Equation (1), the original ColBERT paper uses Cosine similarity for the reranking setting but uses L2 similarity for the end-to-end setting. The choice of the similarity function, Cosine or L2-based, is important for dense retrieval, as it directly affects the scoring of the query and document vectors. However, the original paper makes no experimental assessment of their impact on retrieval effectiveness. In this section, we evaluate the performance of our trained ColBERT models with different similarity functions not only on MSMARCO Dev Small query set but also on the TREC 2019 & 2020 query sets.

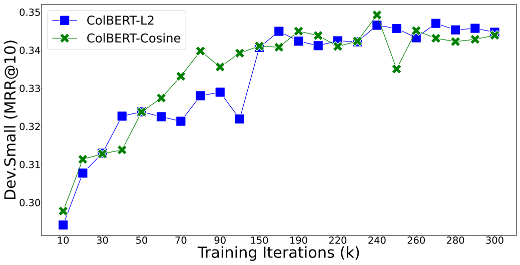

Figure 1: Validation of ColBERT checkpoints on MSMARCO Dev Small in terms of MRR@10.

Results of RQ1.2: Table 2 and Figure 1 compare the retrieval performance of our trained ColBERT models with L2 similarity as well as Cosine similarity functions. During training, from Figure 1, we see that no marked differences in the Dev Small validation queries were observed. During inference, the evaluation results from Table 2 on MSMARCO Dev Small show comparable results for ColBERT models with different similarity methods for reranking but ColBERT-L2 gives slightly higher performance than ColBERT-Cosine under end-to-end settings. We also observe that there is no significant difference between the two similarity methods for reranking TREC 2019 and 2020 queries. However, for end-to-end retrieval results, we find that L2-based similarity exhibits higher performance than Cosine similarity across all metrics on TREC 2019 queries, as well as on Recall@1k on TREC 2020 queries.

Answer to RQ1.2: Overall, we find that (i) there is no significant difference between the two similarity functions for reranking; (ii) for end-to-end retrieval, the L2 similarity shows higher performance than the Cosine similarity function.

3.4 RQ1.3: Training Iterations (with batch size set as 32) Needed for ColBERT Training

The ColBERT paper [13] suggests training all the ColBERT models to 200k iterations with a batch size of 32, but provides no validation guidance for the early stop of the training. The ColBERT code repository[4] suggests performing validation on the saved checkpoints based on their top k reranking performance before indexing. Here,

[3] https://microsoft.github.io/msmarco/

[4] https://github.com/stanford-futuredata/ColBERT/tree/colbertv1

we perform the validation in the reranking setup for both models the official top-1000 BM25 documents using 1000 MSMARCO Dev Small queries, using the MRR@10 metric.

Results of RQ1.3: Figure 1 presents the validation results on the 1000 MSMARCO Dev Small queries. We find that the training process for ColBERT models becomes steady after 150k iterations but ColBERT-Cosine achieves the highest validation performance at around 240k iterations. However, for ColBERT-L2, the validation curve reaches its highest validation performance at 280k iterations.

Answer to RQ1.3: We find that ColBERT training converges at around 150k iterations with batch size as 32. Further training leads to slight improvements in terms of MRR@10 validation scores for both ColBERT-L2 and ColBERT-Cosine models. However, to make a fair comparison to the original paper, we report the performance of all ColBERT models trained with 200k iterations. To this end, we conclude that we can reproduce the results of ColBERT and obtain new findings from our ablation studies. In the next section, we will begin to examine the replicability of ColBERT.

4 REPLICABILITY: FROM COLBERT TO COL★

Besides the reproducibility study of ColBERT, we further look into the replicability of ColBERT by generalising the BERT-based contextualised late interaction mechanism upon various different pretrained models, thus forming Col★ models. Accordingly, we pose our research question for the replicability study as follows: **RQ2:** How does the retrieval effectiveness vary across different contextualised late interaction models?

More specifically, the characteristics of the Col★ models we introduce are summarised in Table 3. The models can be classified within three families, according to the tokenisation technique each model uses, namely WordPiece, BPE and SentencePiece. From the table, we can see that the different base models have different vocabulary sizes and the number of parameters. Moreover, their corresponding ColBERT-like dense indices vary considerably in size.

For the models with the WordPiece tokeniser, we apply the late interaction upon six BERT models with various sized pretrained models, from BERT-Tiny to BERT-Large. The aim of training these variants is to investigate the impact of the number of parameters of the base model that ColBERT encoders are initialised from. In addition, for WordPiece tokeniser models, we also apply ColminiLM and ColELECTRA models. miniLM [34] is a distilled variant of BERT, which aims to reduce the huge number of parameters while retaining BERT's performance. In our work, we use miniLM as a base model for the late interaction dense retrieval mechanism and use $m = 32$ component embeddings. This thus represents a ColBERT-like setting with minimal time- and space-efficiency overheads [1]. We denote this as ColminiLM. Moreover, ELECTRA [3] has been shown to achieve higher performance than a similar-sized BERT on certain NLP tasks and can be implemented as an effective cross-encoder for reranking [7, 21], but its performance has yet to be ascertained for dense retrieval. We implement the late interaction based on ELECTRA and denote this as ColELECTRA.

Secondly, to consider the BPE tokeniser, we train ColRoBERTa with both Base and Large sizes. RoBERTa [20] employs the same model architecture as BERT but exploits the BPE tokeniser, with an increased vocabulary size wrt. ColBERT and ColminiLM. We note that RoBERTa is used as the base model for the ANCE dense retrieval model [38]. We extend the ColBERT model using RoBERTa base model within its BPE tokeniser, denoted as ColRoBERTa.

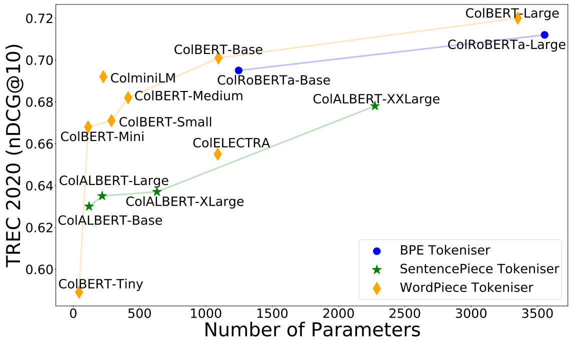

Figure 2: The retrieval effectiveness (y-axis: nDCG@10) of Col★ models on TREC 2020 query set. The x-axis shows the number of parameters of the Col★ models. Different markers indicate the tokenisation technique used by the Col★ models.

Similar to miniLM [34], ALBERT [15] aims at reducing the number of parameters of BERT by sharing parameters across transformer layers. In our replicability experiments, we train four ColALBERT models by fine-tuning various sized base models, including 'Base', 'Large', 'XLarge' and 'XXLarge' ALBERT models. ColALBERT models employ the SentencePiece tokeniser, which allows us a third tokeniser setting.

All the Col★ models listed in Table 3 are trained following the original ColBERT training setup, with a batch size of 32 and the query length and document length are set as 32 and 180, respectively. Table 3 also provides salient details and statistics of the models and their corresponding indices. In addition, for all the Col★ models, except ColminiLM, we fine-tune the models upto 300k iterations, selecting the final model based on reranking effectiveness on the 2019 queries. For ColminiLM, we use the checkpoint `vespa-engine/col-minilm` provided by the author of [1] which was trained similarly. Since using the MSMARCO Dev query set for validation is computationally expensive, we used a smaller set of TREC 2019 queries for validation instead. All the Col★ models are trained with the Cosine similarity method.

Figure 2 shows the number of parameters and the tokeniser's impact on the retrieval effectiveness of various Col★ models. An ANOVA study indicates that both the number of parameters and the type of tokeniser used have a significant impact on the nDCG@10 scores, at a significance level of $p < 0.05$. The performance of the models on natural language understanding tasks tends to improve with an increase in the number of trainable parameters [11], although this is not always the case [40]. Our findings, as displayed in Figure 2, indicate that for BERT-based, ALBERT-based and RoBERTa-based Col★ models, retrieval effectiveness tends to increase with an increase in the number of parameters. It should be noted that larger parameterised models may be more prone to overfitting and require more computational resources for both training and inference. Additionally, the quality of the training data and the model architecture can also impact the retrieval performance of Col★ models. More importantly, considering the environmentally friendly information retrieval [30], we focus on the Col★ models with different tokenisation techniques and investigate the impact

Table 3: Characteristics for different Col★ models with contextualised late interaction.

Col★ Model	Tokeniser	Vocab. Size	Index size	Embedding Dim.	Number of Parameters	HF Base Model
ColBERT-Base	WordPiece	30,522	373G	128	1095M	bert-base-uncased
ColBERT-Large	WordPiece	30,522	-	128	3353M	bert-large-uncased
ColBERT-Tiny	WordPiece	30,522	-	128	44M	bert-tiny
ColBERT-Mini	WordPiece	30,522	-	128	112M	bert-mini
ColBERT-Small	WordPiece	30,522	-	128	288M	bert-small
ColBERT-Medium	WordPiece	30,522	-	128	414M	bert-medium
ColELECTRA-Base	WordPiece	30,522	-	128	1090M	electra-small-discriminator
ColminiLM	WordPiece	30,522	64G	32	227M	-
ColRoBERTa-Base	BPE	50,267	356G	128	1247M	roberta-base
ColRoBERTa-Large	BPE	50,267	-	128	3555M	roberta-large
ColALBERT-Base	SentencePiece	30,002	199G	128	119M	albert-base-v2
ColALBERT-Large	SentencePiece	30,002	-	128	218M	albert-large-v2
ColALBERT-XLarge	SentencePiece	30,002	-	128	631M	albert-xlarge-v2
ColALBERT-XXLarge	SentencePiece	30,002	-	128	2275M	albert-xxlarge-v2

of the tokenisation techniques on semantic matching behaviour. To this end, we select to index ColBERT-Base, ColRoBERTa-Base and ColALBERT-Base models. We also compare with the ColminiLM model, which reduces the embedding dimension from 128 to 32.

4.1 RQ2: Retrieval Effectiveness across Col★?

To understand if the de-facto BERT base model can be replaced for implementing the late interaction mechanism, we deploy the late interaction technique on various contextualised pretrained language models (which also use varying tokenisers).

Results of RQ2: Table 4 reports the replication results for the selected Col★ models for both the reranking and end-to-end dense retrieval scenarios on both TREC DL 2019 and 2020 query sets.

First, we analyse the ColminiLM model, which exploits a lightweight BERT model and uses the identical WordPiece tokeniser as ColBERT. From the reranking results in Table 4, we see that ColminiLM significantly outperforms BM25 and shows comparable performance to ColBERT across the metrics on both TREC 2019 and 2020 query sets, except markedly lower than ColBERT in terms of nDCG@10 on TREC 2019 queries. Similarly, for the end-to-end retrieval experiments, ColminiLM exhibits significant improvements over BM25. However, compared to ColBERT, ColminiLM shows significantly lower MAP, nDCG@10 and Recall on TREC 2019 and significantly lower MAP and Recall on TREC 2020 queries. The lower performance of ColminiLM can be explained in that, as shown in Table 3, it requires much fewer parameters (only 20% of the ColBERT parameters). ColminiLM remains promising as it shows comparable nDCG@10 performance on the test queries (TREC 2020) and it has a smaller index size (~17% of the ColBERT index size).

Next, we analyse ColRoBERTa. We find that ColRoBERTa exhibits comparable retrieval effectiveness to ColBERT and markedly improvements over BM25 when employed as a reranker on top of the BM25 sparse retrieval across all the reported metrics. In addition, it shows comparable performance wrt. ColBERT in the dense end-to-end retrieval scenario on TREC 2019 and 2020 queries, except MAP on TREC 2020 query set. Overall, we find that ColRoBERTa is a good replacement of ColBERT.

For ColALBERT, we observe that it shows lower performance than ColBERT across all the reported metrics on both reranking and end-to-end dense retrieval implementations on both query sets. Similar to ColminiLM, ColALBERT has significantly fewer parameters and a simplified model structure than ColBERT. Overall, ColALBERT has low performance in terms of the precision metrics: MAP, nDCG@10 and MRR@10, and surprisingly high performance in terms of the Recall@1k.

Finally, it is notable that, at least on this query set, the other model families consistently do not outperform the BERT family. This suggests that more recent families of pretrained language model (ALBERT, RoBERTa) have not equated to improvements in a downstream retrieval task compared to the original BERT model.

Answer to RQ2: We conclude that we can replicate the contextualised late interaction mechanism upon various pretrained models. More specifically, we find that, when compared to the ColBERT model, ColRoBERTa exhibits a competitive performance to ColBERT. However, consistent with the findings from Figure 2, we find that the ColminiLM and ColALBERT models show slightly lower retrieval effectiveness than ColBERT due to their lightweight model structures. Notably, no model family exceeds BERT in terms of effectiveness for a comparable number of parameters.

5 INSIGHTS: SEMANTIC MATCHING

The success of reproducibility and replicability of ColBERT motivates us to investigate the semantic matching behaviour to generate more insights. Thus, to examine more deeply how the different contextualised late interaction models perform retrieval, we turn to investigate their semantic matching behaviour. In particular, in this section, we introduce an approach to measure the semantic contribution to relevance scoring of documents with contextualised late interaction models. Then, we conduct experiments to address the following research questions: **RQ3.1:** How does the semantic matching behaviour vary across different contextualised late interaction models? (Section 5.1) **RQ3.2:** Can we characterise the salient token families of matches, i.e., which type of tokens contribute the most to semantic matching? (Section 5.2) and **RQ3.3:** Can we quantify the contribution of different types of matching behaviour, namely the lexical match and semantic match as well as special token match, to the retrieval effectiveness? (Section 5.3)

Figure 3 illustrates the contextualised late interaction mechanism among a query and a document for ColBERT (left) and ColRoBERTa (right) models. For every query token, on the columns, a `X` marks the matching document tokens with the highest similarity score, hence contributing to the final relevance score, as in Equation (1). For ColBERT, query tokens such as `the`, `w`, and `##w` exact match with lexically identical document tokens. At the same time, semantic matching behaviour occurs with the query tokens `why` and `enter`, matching with document tokens `because` and `entered`, respectively. However, the late interaction for ColRoBERTa produces

Table 4: Performance of contextualised late interaction models. The † (⋄) symbol denotes statistically significant differences compared to BM25 (ColBERT). The highest value in each column is boldfaced.

Models	TREC DL 2019					TREC DL 2020				
	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP	MAP@1k	nDCG@10	MRR@10	R@1k	Mean SMP
BM25 (PyTerrier)	0.286	0.480	0.640	0.755	-	0.293	0.494	0.615	0.807	-
BM25 » Late Interaction										
ColBERT	**0.459**†	**0.713**†	0.847†	0.755	0.375	**0.484**†	**0.707**†	0.835†	0.807	0.387
ColminiLM	0.431†	0.654†⋄	0.811†	0.755	0.362	0.458†	0.685†	**0.866**†	0.807	0.363
ColRoBERTa	0.458†	0.695†	**0.865**†	0.755	**0.599**	0.462†	0.695†	0.844†	0.807	**0.607**
ColALBERT	0.412†⋄	0.634†⋄	0.821†	0.755	0.367	0.401†⋄	0.630†⋄	0.751†	0.807	0.390
ANN Search » Late Interaction										
ColBERT	**0.445**†	**0.708**†	0.857†	**0.773**	0.390	**0.473**†	**0.690**†	0.832†	**0.806**	0.406
ColminiLM	0.388†⋄	0.631†⋄	0.811†	0.698⋄	0.382	0.434†⋄	0.672†	**0.860**†	0.762⋄	0.388
ColRoBERTa	0.426†	0.684†	**0.866**†	0.738	**0.610**	0.423†⋄	0.666†	0.828†	0.760	**0.622**
ColALBERT	0.356⋄	0.613†⋄	0.769	0.772	0.381	0.367†⋄	0.604†⋄	0.745†	0.792	0.413

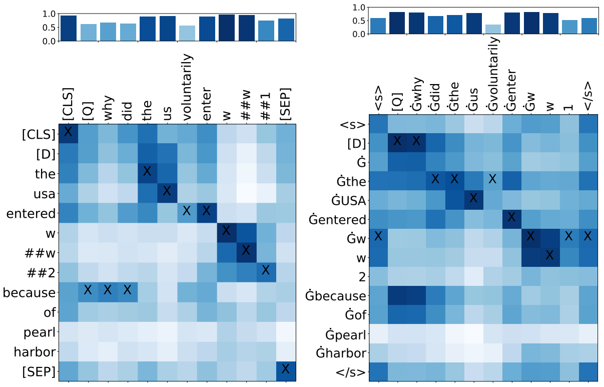

Figure 3: Late interaction diagrams for ColBERT and ColRoBERTa models between the query: *why did the us voluntarily enter ww1* and the document: *the usa entered ww2 because of pearl harbor*. For each column, the heatmap indicates the similarity scores among all the document embeddings for each query embedding, where the highest similarity score is highlighted with the symbol X. The top histogram depicts the magnitude of the contribution of the maximum similarity of each query embedding for the final relevance score between the query and document. The [MASK] tokens are omitted.

different token forms and different lexical and semantic matches with document tokens and some query tokens. Thus, we observe that the base model and the tokenisation algorithm not only affect the model size (c.f. Table 3), but, more importantly, they impact the way the matching between queries and documents is conducted within the late interaction mechanism.

Indeed, a benefit of late interaction over multiple contextualised dense representations is that we can investigate the lexical and semantic matching behaviour. To do so, we use a recently proposed technique to quantify the extent of the semantic matching during the contextualised late interaction mechanism [36, 37]. More specifically, among the query-document token pairs contributing to the similarity score computation in late interaction, *lexical matching* corresponds to the contributing pairs with identical tokens, while *semantic matching* corresponds to different contributing query-document tokens pairs (e.g. why with because). Formally, given a query q and the list R_k of the top-ranked k passages, the *Semantic Match Proportion* (SMP) at rank cutoff k wrt. q and R_k is defined as:

$$\mathrm{SMP}(q, R_k) = \sum_{d \in R_k} \frac{\sum_{i \in \mathrm{toks}(q)} \mathbb{1}[t_i \neq t_j] \cdot \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}}{\sum_{i \in \mathrm{toks}(q)} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}}, \qquad (2)$$

where toks(q) contains the indices of the query embeddings that correspond to the tokens produced by the tokeniser, and t_i and t_j denote the token ids of the i-th query embedding and j-th passage embedding, respectively. Special tokens such as [SEP], [CLS], [Q] and [MASK] for the WordPiece-based models always match semantically. Therefore, we exclude these special tokens when measuring the SMP value for a model. However, we revisit and quantify their contribution to the retrieval effectiveness in Section 5.3.

5.1 RQ3.1: Semantic Matching Behaviour of Col★

Results of RQ3.1: Now, we analyse the semantic matching proportion scores for the selected Col★ models presented in Table 4. For all the compared models, we report the Mean SMP values computed at rank cutoff $k = 10$. From the Mean-SMP columns in Table 4, we find that ColminiLM, with the same tokenisation and vocabulary size of ColBERT, shows a similar, but slightly reduce semantic matching behaviour to ColBERT. In addition, the SentencePiece tokeniser also shows comparable semantic matching scores. Finally, ColRoBERTa performs more of its matching in the semantic space, both for the reranking and dense retrieval scenarios. This is actually not in line with our expectations – indeed, with a larger vocabulary, we expected to see more exact matches by ColRoBERTa. We explain further the ColRoBERTa's behaviour in the next section.

Answer to RQ3.1: Overall, we observe that using different tokenisers, Col★ exhibits different amounts of semantic matching. In particular, the BPE tokeniser based ColRoBERTa model exhibits a stronger preference for semantic matching compared to WordPiece and SentencePiece tokeniser based models. Based on the findings of RQ3.1, we next inspect how semantic matching proportion values can be attributed to different families of tokens (Section 5.2), and to determine the contribution of lexical vs. semantic matching types to retrieval effectiveness (Section 5.3).

5.2 RQ3.2: SMP on Salient Token Families

We now further deepen our analysis on the internals of the late interaction mechanism, by investigating the semantic matching

Table 5: Salient token families of query (Q) and document (Doc) tokens.

	Notation	Type of Tokens	Example
Q	QuesToken	Question tokens	who, what, where, when, why, which, and how
Doc	SubToken	Sub-word tokens	Tokens beginning with ## for ColBERT and ColminiLM, not beginning with space for ColRoBERTa, and not beginning with _ for ColALBERT
	SwToken	Stopwords tokens	Terrier stopwords such as `is` and `a`
	NumToken	Numeric tokens	Token corresponding to single-digit numbers
	StemToken	Stemmed tokens	Tokens in the same form as the matching query token after applying Porter stemming
	Low$_{idf}$Token	Low IDF tokens	Tokens with IDF below the 25th percentile of IDF distribution
	Med$_{idf}$Token	Medium IDF tokens	Tokens with IDF between the 25th and the 75th percentiles of IDF distribution
	High$_{idf}$Token	High IDF tokens	Tokens with IDF above the 75th percentile of IDF distribution

Table 6: Mean semantic matching proportion for the salient document token families in query and document on TREC DL 2020. The highest value among the salient token families in each column is boldfaced.

		BM25 (PyTerrier) » Late Interaction				ANN Search » Late Interaction			
		ColBERT	ColminiLM	ColRoBERTa	ColALBERT	ColBERT	ColminiLM	ColRoBERTa	ColALBERT
	All Types	0.387	0.363	0.607	0.390	0.406	0.388	0.622	0.413
Q	QuesToken	0.085	0.087	0.090	0.067	0.087	0.089	0.091	0.070
Doc	SubToken	0.009	0.011	0.126	0.179	0.013	0.020	0.133	0.190
	SwToken	0.163	0.127	0.159	0.125	0.169	0.134	0.165	0.130
	NumToken	0.017	0.018	0.003	0.001	0.019	0.018	0.004	0.001
	StemToken	0.022	0.024	0.025	0.019	0.023	0.022	0.026	0.020
	Low$_{idf}$Token	**0.365**	**0.344**	**0.517**	**0.270**	**0.381**	**0.361**	**0.523**	**0.289**
	Med$_{idf}$Token	0.021	0.018	0.068	0.018	0.025	0.026	0.074	0.018
	High$_{idf}$Token	0.001	0.001	0.005	0.004	0.001	0.001	0.006	0.005

contribution of individual query and document tokens. To this end, we identify salient families of tokens in queries and documents, based on our intuitions about how contextualised embeddings are matched. Table 5 summarises the identified token families.

Results of RQ3.2: We inspect the semantic matching behaviour for different contextualised late interaction models with various salient token families listed in Table 5. More specifically, we are more concerned about what matching behaviour is performed for the question tokens in the query and seven families of salient tokens in the document. Table 6 presents the semantic matching proportion scores for the above salient token families for all four contextualised late interaction models. We examine the semantic matching behaviour for both the reranking and end-to-end dense retrieval scenarios on the TREC DL 2020 query set. From Table 6, we find that question tokens occurring in the query exhibit low semantic matching scores. Among all the families of salient tokens from documents, semantic matching prefers the low IDF (i.e. frequent) tokens, followed by the family of stopwords tokens. However, semantic matching seldom occurs in the medium and high IDF tokens, which means such rare tokens are more likely to exactly match during scoring. In addition, token families include stemmed, numeric and sub-word tokens all exhibiting low semantic matching proportion values. Finally, comparing the different Col⋆ models, we find that ColRoBERTa exhibits the highest semantic matching proportion scores, which is consistent with the findings obtained in Table 4. More interestingly, although ColBERT, ColminiLM and ColALBERT show similar SMP values overall for all types of tokens in Table 4, results in Table 6 indicate that for their semantic matching occurs for different types of tokens. For instance, ColBERT and ColminiLM tend to perform semantic matching for the tokens with relatively low IDF scores and sub-word tokens. ColALBERT (SentencePiece) behaves more similarly to the WordPiece-based models (ColBERT & ColminiLM), except that it more semantic matching comes from sub-word tokens and less from low-IDF tokens.

Interestingly, ColRoBERTa exhibits the highest semantic matching, mostly on low IDF (i.e. frequent) tokens. For different models, we inspect queries returning the same documents, and we focus on those with different matching proportions for the same document, and we explain these differences as follows: as RoBERTa's vocabulary is case-sensitive, some words can be represented by a whole token when occurring in lower-case, but resort to sub-word tokens when starting with an uppercase letter (see `Casualties` vs. `casualties` examples in the last row of Table 1). To make a match between these words requires a semantic match (involving relatively frequent sub-word tokens), where a case-insensitive model would have made an exact match (that would likely have been easier to learn). Indeed, the original RoBERTa authors acknowledged that their tokenisation configuration choice might not be the most effective [20]. This analysis indicates the challenges for the search with case-sensitive contextualised language models.

Answer to RQ3.2: Overall, in quantifying the extent of semantic matching for various token families, we find that low IDF tokens are most likely to exhibit semantic matching. In the next section, we conduct further experiments to quantify the contribution of different types of matching to retrieval effectiveness.

5.3 RQ3.3: Contribution of Matching Types to Retrieval Effectiveness

Finally, as the final outcome of matching behaviour is the ranking of the document, we analyse how the final retrieval effectiveness

Table 7: Impact of different types of matching behaviour for TREC DL 2020 on nDCG@10, and relative decrease from All (Δ). The † and ⋄ symbols denote statistically significant differences compared to the BM25 and the all types matching of a model. The highest nDCG@10 value in each column is boldfaced.

Models	**All Types**	**Lexical Matching**		**Semantic Matching**		**Special Token Matching**	
	nDCG@10	nDCG@10	Δ	nDCG@10	Δ	nDCG@10	Δ
BM25 (PyTerrier)	0.4936	-	-	-	-	-	-
			BM25 (PyTerrier) » Late Interaction				
ColBERT	**0.707**†	**0.527**⋄	-25.5%	0.139†⋄	-80.3%	0.519⋄	-26.6%
ColminiLM	0.685†	0.487⋄	-28.8%	0.074†⋄	-89.1%	0.523⋄	-23.7%
ColRoBERTa	0.695†	0.397†⋄	-42.9%	**0.261**†⋄	-62.5%	**0.635**†⋄	-8.6%
ColALBERT	0.630†	0.505⋄	-19.8%	0.074†⋄	-88.2%	0.460⋄	-27.1%
			ANN Search » Late Interaction				
ColBERT	**0.690**†	**0.492**⋄	-28.7%	0.002†⋄	-99.7%	0.384⋄	-44.4%
ColminiLM	0.672†	0.426⋄	-36.6%	0.001†⋄	-99.9%	0.347†⋄	-48.4%
ColRoBERTa	0.666†	0.350†⋄	-47.5%	**0.157**†⋄	-76.4%	**0.574**⋄	-13.8%
ColALBERT	0.604†	0.411⋄	-32.0%	0.007†⋄	-98.8%	0.341†⋄	-43.4%

correlates with the lexical matches and the semantic matches. To conduct this ablation, we also consider retrieval using only "special" tokens, such as `[CLS]` and `[Q]`, which always match semantically.

Results of RQ3.3: Now, we examine the retrieval effectiveness by conducting only special matching, only semantic matching, as well as special token matching (e.g., `[CLS]`, `[Q]`, `[SEP]` and `[MASK]` tokens for WordPiece tokeniser), in response to the input queries of the TREC DL 2020 query set. Table 7 presents the impact of performing a particular type of matching on the retrieval effectiveness (measured by nDCG@10) as well as the reduction percentage compared to all types of matching. From Table 7, we find that performing each type of matching alone results in significant reductions in effectiveness compared to all types of matching, for both the reranking and end-to-end dense retrieval scenarios. In particular, for all models except ColRoBERTa, lexical matching contributes to the highest retrieval effectiveness; for ColRoBERTa, the special tokens have excellent effectiveness (contributing 80-90% of the full effectiveness). Similarly, semantic matching alone exhibits low effectiveness but is strongest for ColRoBERTa (this again demonstrates the strong semantic properties of the ColRoBERTa embeddings). Moreover, Table 5 tells us that this semantic matching is mostly concentrated on frequent (low IDF) tokens. Finally, the high performance of lexical matching is mostly related to medium and high IDF tokens - indeed, this observation echoes the finding of [5] that ColBERT is able to capture more important terms by performing exact matches. Our work systematically quantifies and generalises this finding to various contextualised late interaction models. However, different types of matching need to work together to achieve optimal retrieval effectiveness, as performing any type of matching alone will result in a significant drop in retrieval effectiveness compared to performing all types of matching.

Answer to RQ3.3: We find that the late interaction mechanism benefits more from lexical matching than semantic matching. In addition, special tokens, such as the `[CLS]` token, play a very important role in matching, especially for the ColRoBERTa model.

6 CONCLUSIONS

This work provides a comprehensive study that investigates the reproducibility and replicability of ColBERT and sheds insights into the semantic matching behaviour in multiple representation dense retrieval. Our main findings and insights are summarised as follows:

Based on the **Reproducibility** experiments of ColBERT, we are able to successfully reproduce the performance of ColBERT on various query sets. In addition, several ablation studies show that more training interactions still help improve the retrieval effectiveness of ColBERT. The L2 similarity function gives higher performance than Cosine for the end-to-end setting and exhibits comparable performance for the reranking retrieval.

For **Replicability**, we extend ColBERT to Col★ by implementing the contextualised late interaction mechanism upon various pretrained models with different tokenisers. We find that the base pretrained model used for ColBERT can greatly impact the retrieval performance, but models from the BERT family are the most effective.

Finally, we conduct the **Insights** experiments to explore more useful insights behind the contextualised late interaction. In particular, we introduce a metric to quantify semantic matching for dense retrieval. Extensive experimental results reveal that: (i) Col★ models with different tokenisation methods show different semantic matching values, in particular, the ColRoBERTa model exhibits higher SMP values due to its case-sensitive tokeniser; (ii) among various salient families of tokens, low IDF and stopwords tokens are more likely to perform semantic matching; (iii) performing only exact matching and only special token matching contribute more than only semantic matching to all types matching retrieval effectiveness. Overall, our experimental results explain how ColBERT-like models perform retrieval, and can shed insight into more effective dense retrieval model design.

ACKNOWLEDGMENTS

This work is supported, in part, by the spoke "FutureHPC & BigData" of the ICSC – Centro Nazionale di Ricerca in High-Performance Computing, Big Data and Quantum Computing funded by European Union – NextGenerationEU, and the FoReLab project (Departments of Excellence). Xiao Wang acknowledges support by the China Scholarship Council (CSC) from the Ministry of Education of P.R. China. We thank Jo Kristian Bergum for assistance with the ColminiLM checkpoint.

REFERENCES

[1] Jo Kristian Bergum. 2021. Pretrained Transformer Language Models for Search - part 4. https://blog.vespa.ai/pretrained-transformer-language-models-for-search-part-4/

[2] Kaj Bostrom and Greg Durrett. 2020. Byte Pair Encoding is Suboptimal for Language Model Pretraining. In *Proceedings of EMNLP: Findings*. 4617–4624.

[3] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceddings of ICLR*.

[4] 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.

[5] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. A White Box Analysis of ColBERT. In *Proceedings of ECIR*. 257–263.

[6] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2022. Match your words! a study of lexical matching in neural information retrieval. In *Proceedings of ECIR*. 120–127.

[7] Mitko Gospodinov, Sean MacAvaney, and Craig Macdonald. 2023. Doc2Query: When Less is More. In *Proceedings of ECIR*.

[8] Weidong Guo, Mingjun Zhao, Lusheng Zhang, Di Niu, Jinwen Luo, Zhenhua Liu, Zhenyang Li, and Jianbo Tang. 2021. LICHEE: Improving Language Model Pre-training with Multi-grained Tokenization. In *Proceedings of ACL-IJCNLP: Findings*. 1383–1392.

[9] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving efficient neural ranking models with cross-architecture knowledge distillation. In *arXiv preprint arXiv:2010.02666*.

[10] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of SIGIR*. 113–122.

[11] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

[12] 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.

[13] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of SIGIR*. 39–48.

[14] Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of EMNLP*. 66–71.

[15] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceddings of ICLR*.

[16] Carlos Lassance, Maroua Maachou, Joohee Park, and Stéphane Clinchant. 2022. Learned Token Pruning in Contextualized Late Interaction over BERT (ColBERT). In *Proceedings of SIGIR*. 2232–2236.

[17] Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to Train Your DRAGON: Diverse Augmentation Towards Generalizable Dense Retrieval. *arXiv preprint arXiv:2302.07452* (2023).

[18] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2020. Distilling dense representations for ranking using tightly-coupled teachers. *arXiv preprint arXiv:2010.11386* (2020).

[19] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on RepL4NLP*. 163–173.

[20] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. RoBERTa: A Robustly Optimized BERT Pretraining Approach. In *Proceddings of ICLR*.

[21] Sean MacAvaney, Nicola Tonellotto, and Craig Macdonald. 2022. Adaptive re-ranking with a corpus graph. In *Proceedings of CIKM*. 1491–1500.

[22] Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2021. On Single and Multiple Representations in Dense Passage Retrieval. *IIR 2021 Workshop* (2021).

[23] Suraj Nair, Eugene Yang, Dawn Lawrie, Kevin Duh, Paul McNamee, Kenton Murray, James Mayfield, and Douglas W Oard. 2022. Transfer learning approaches for building cross-language dense retrieval models. In *roceddings of ECIR*. 382–396.

[24] Association of Computing Machinery. 2020. Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-and-badging-current.

[25] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of NAACL*. 5835–5847.

[26] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

[27] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* (2020), 1–67.

[28] Jenalea Rajab. 2022. Effect of Tokenisation Strategies for Low-Resourced Southern African Languages. In *Proceedings of Workshop on African Natural Language Processing*.

[29] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2021. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of NAACL*.

[30] Harrisen Scells, Shengyao Zhuang, and Guido Zuccon. 2022. Reduce, Reuse, Recycle: Green Information Retrieval Research. In *Proceedings of SIGIR*. 2825–2837.

[31] Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *Proceddings of ICASSP*. IEEE, 5149–5152.

[32] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of ACL*. 1715–1725.

[33] Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2022. Impact of Tokenization on Language Models: An Analysis for Turkish. *arXiv preprint arXiv:2204.08832* (2022).

[34] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceddings of NeurIPS*, Vol. 33. 5776–5788.

[35] Xiao Wang, Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2022. An Inspection of the Reproducibility and Replicability of TCT-ColBERT. In *Proceedings of SIGIR*. 2790–2800.

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

[37] Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2022. ColBERT-PRF: Semantic Pseudo-Relevance Feedback for Dense Passage and Document Retrieval. *ACM Transactions on the Web* (2022).

[38] 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*.

[39] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of SIGIR*. 1503–1512.

[40] Ruiqi Zhong, Dhruba Ghosh, Dan Klein, and Jacob Steinhardt. 2021. Are Larger Pretrained Language Models Uniformly Better? Comparing Performance at the Instance Level. In *Proceedings of ACL: Findings*. 3813–3827.