Enhancing ColBERT: A Method for Reducing Space Complexity and Accelerating Retrieval Speed

Hai Nguyen T., Huong Le T.

Proceedings of the 38th Pacific Asia Conference on
Language, Information and Computation (PACLIC 38)

Nathaniel Oco, Shirley N. Dita, Ariane Macalinga Borlongan, Jong-Bok Kim (eds.)

2024

Enhancing ColBERT: A Method for Reducing Space Complexity and Accelerating Retrieval Speed

Hai Nguyen T. and **Huong Le T.** *
School of Information and Communication Technology,
Hanoi University of Science and Technology, Hanoi, Vietnam
trunghainguyenhp02@gmail.com and huonglt@soict.hust.edu.vn

Abstract

Recent advancements in neural information retrieval systems have focused on optimizing efficiency and effectiveness using BERT-based models for semantic encoding. The ColBERT model's late-interaction mechanism, while effective, leads to larger indexes and slower retrieval speeds compared to single-vector approaches. This study introduces a pruning method for ColBERT vector embeddings, utilizing a small network to assign weights to essential tokens for scoring and eliminating less significant ones with a threshold. Our method significantly reduces space requirements and enhances retrieval speed, with a minimal decrease in performance, as demonstrated using a Vietnamese Wikipedia-based dataset.

1 Introduction

Information retrieval (IR) has been a significant area of research within Natural Language Processing (NLP) for a long time. Traditional IR methods, such as sparse retrievers (e.g., BM25), are now being outperformed by dense neural retrievers that use deep learning models to calculate similarity scores between queries and documents. Recent advancements in pre-trained language models (PLMs) based on the Transformer architecture (Vaswani et al., 2017) have significantly improved neural IR techniques, boosting performance across various benchmarks.

Neural IR paradigms mainly differ in their scoring mechanisms. Cross-Encoder architectures leverage self-attention across all tokens in a query-passage pair to generate similarity scores, yielding superior ranking performance with fine-grained, contextualized embeddings. In contrast, Single-Vector Bi-Encoders create single-vector representations for each query and document by employing pooling mechanisms, with similarity assessed using metrics like cosine similarity. These models, trained with contrastive learning and sophisticated pre-training procedures, effectively capture semantic nuances and offer high performance and efficiency, enabling rapid retrieval via pre-indexing and efficient nearest neighbor search.

However, both Cross-Encoders and Single-Vector Bi-Encoders have their limitations. Cross-Encoders are not scalable for real-world applications because they require processing both the query and the document through a large language model for every pair, preventing search time reduction through precomputation. Single-Vector Bi-Encoders, while allowing faster searches, often provide lower ranking effectiveness and struggle to encapsulate the semantics of entire documents for some datasets (Sciavolino et al., 2021).

To balance efficiency with contextual depth in IR, the ColBERT model (Khattab and Zaharia, 2020) employs token embeddings from PLMs and a late interaction technique to calculate query-document similarity scores. ColBERT uses multiple embeddings per document, capturing complex semantic relationships and outperforming most Single-Vector Bi-Encoder and some Cross-Encoder models. However, this comes at the cost of increased computation and storage requirements, potentially exceeding RAM capacities and affecting retrieval speed on limited hardware.

This research introduces a novel token pruning method for ColBERT to reduce the number of vectors stored by ColBERT with minimal performance trade-offs. We propose directly learning the importance of tokens in documents via a neural network layer during training and use this layer to assign weights to tokens based on their relevance, keeping only the important ones (tokens with high weights) when indexing. This approach effectively preserves document keywords, significantly reducing storage requirements and improving retrieval speed.

In summary, our contributions include:

* Corresponding author: huonglt@soict.hust.edu.vn

1. We introduce a novel token pruning method for ColBERT that achieves a balance between efficiency and effectiveness, enabling flexible adjustment of the pruning threshold.

2. We propose an effective training approach for the weight-assigning neural network, utilizing distillation from a well-trained model and supervision from token classification tasks such as NER and POS.

3. Our method is evaluated on a Vietnamese Wikipedia-based dataset, and we compare it with other research aimed at improving the efficiency of ColBERT.

2 Background & Related Works

2.1 Neural Information Retrieval

As data grows, traditional match-based search methods are becoming less effective, prompting a shift toward semantic search. The Transformer architecture (Vaswani et al., 2017) and advanced language models (Devlin et al., 2019; Liu et al., 2019) have established neural retrieval as the dominant approach, leading to the development of numerous models (Karpukhin et al., 2020; Nogueira and Cho, 2020; Formal et al., 2021).

Among these neural models, deep interaction-based models known as cross-encoders (Nogueira and Cho, 2020; Dai and Callan, 2019; Phan and Le, 2023) achieve high retrieval effectiveness but at the expense of speed. Despite efforts to reduce their latency (MacAvaney et al., 2020; Gao et al., 2020), these models remain impractical for real-world applications and are typically reserved for re-ranking after initial retrieval.

In contrast, representation-based models, such as Single-Vector Bi-Encoders (Karpukhin et al., 2020), leverage deep language models to produce single embedding vectors representing documents or queries. These embeddings retain the content and context of the entire input text. Similarity is then computed using simple metrics such as cosine similarity, with retrieval performed by selecting the top results via nearest neighbor search. This approach is widely favored, and many studies have proposed various training methods to create robust embeddings that yield accurate retrieval results (Qu et al., 2021; Xiao et al., 2022; Nguyen and Le, 2023).

2.2 Multi-Vector Bi-Encoder

In addition to learning a single representation, several studies have proposed using multiple representations for queries and documents, coupled with simple interaction mechanisms to compute similarity scores. This approach mitigates the limitations of single-vector representations in terms of accuracy and interpretability.

The Poly-encoder model (Humeau et al., 2019) encodes queries into a set of vectors, and the ME-BERT model (Luan et al., 2021) does the same for documents. Notably, ColBERT (Khattab and Zaharia, 2020) encodes both queries and documents into multiple vectors and employs the MaxSim late interaction mechanism for similarity computation. COIL (Gao et al., 2021), developed concurrently with ColBERT, adopts a similar idea but incorporates hard matching for faster search.

While these models are highly effective, they have higher computational complexity than Single-Vector Bi-Encoders, higher retrieval latency, and require storing numerous vectors. Subsequent studies have focused on improving these aspects by reducing latency and index size through advanced search procedures (Santhanam et al., 2022a), quantization (Santhanam et al., 2022b), and more robust training methods (Santhanam et al., 2022b).

2.3 Pruning for ColBERT

Pruning directly addresses the storage and computational costs of ColBERT by retaining embeddings only for the most important tokens. Recent research on token pruning (Liu et al., 2024; Lassance et al., 2021; Lassance, 2022) proposed heuristics for selecting tokens to retain, such as:

- The first few tokens in a document.
- Tokens with the highest IDF scores.
- Tokens with the highest attention scores.

These heuristics, applied during training or as a post-processing step, have shown effectiveness but are not optimal, often significantly reducing retrieval accuracy.

The ColBERTer model (Hofstätter, 2022) further proposes reducing the number of vectors by using whole-word embeddings and a ReLU gate to filter tokens. Although this method can identify important tokens, it faces challenges in achieving training convergence and lacks flexibility in adjusting the pruning level.

Our research directly learns from data to identify important tokens, aiming to achieve high accuracy while using soft weights for tokens to enable flexible pruning during retrieval.

3 Methodology

In this paper, our primary objective is to minimize the space requirements and enhance the retrieval speed of the ColBERT model. To contextualize our contributions, we begin with a comprehensive overview of the ColBERT model, followed by the introduction of our proposed ColBERT-Kw model, which uses a small network to assign weights to each document token, facilitating effective token pruning based on importance. Since ColBERT-Kw exhibits training difficulties with conventional supervised contrastive learning, we propose a knowledge distillation procedure to improve convergence.

3.1 ColBERT Modelling

ColBERT is a Multi-Vector Bi-Encoder model that uses a pre-trained Language Model (PLM), such as BERT, to independently encode queries and documents into high-dimensional vector embeddings. In this model, a query encoder and a document encoder transform a query Q and a document D into sequences of fixed-size embeddings. The key innovation in ColBERT is its late interaction mechanism, MaxSim, which calculates the maximum similarity for each query token embedding against all document token embeddings. The overall similarity between Q and D is then defined as:

$$s(Q, D) = \sum_{i \in E_Q} \underbrace{\max_{j \in E_D} E_Q[i] \cdot E_D[j]^\intercal}_{\text{MaxSim}} \quad (1)$$

where E_Q and E_D are the sequences of contextualized vector embeddings from Q and D. $E_Q[i]$ and $E_D[j]$ are the vector embeddings of token i in E_Q and token j in E_D, respectively. The intuition behind this mechanism is to align each query token with the most contextually relevant passage token, quantifying these matches and combining the partial scores across the query.

During offline indexing, all vector embeddings for the corpus are precomputed and stored. For retrieval, ColBERT first calculates the query vector embeddings, then performs a nearest neighbor search for all query vectors. Documents with vectors appearing in the top neighbors are then fully scored using MaxSim.

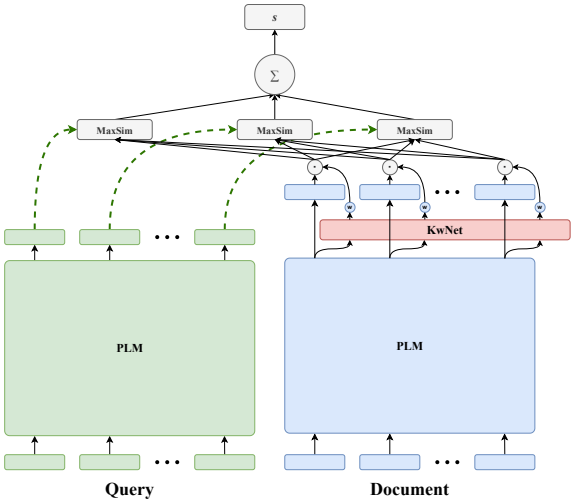

Figure 1: ColBERT-Kw

In this study, we utilize the ColBERT model with a few modifications:

- Normalizing the ColBERT score by dividing it by the query length (number of tokens). This normalization does not affect document ranking but improves model convergence during training:

$$s(Q, D) = \frac{1}{l} \sum_{i \in E_Q} \max_{j \in E_D} E_Q[i] \cdot E_D[j]^\intercal \quad (2)$$

  in which l is the query length.

- Omitting token clipping and augmentation with [MASK] tokens as in the original paper, as they lead to information loss and slight, hard-to-interpret improvements (Giacalone et al., 2024), respectively.

3.2 ColBERT-Kw

We propose the ColBERT-Kw model, where "Kw" denotes "Keyword". This model retains the core components of the standard ColBERT model and introduces an additional network layer named KwNet. KwNet processes contextually enriched token representations from a document, generated by the PLM, and assigns weights to these tokens. The architecture of ColBERT-Kw is illustrated in Figure 1. We implemented KwNet as a

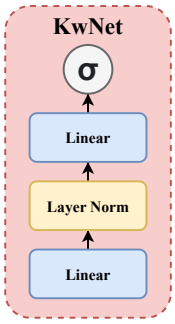

Figure 2: KwNet

simple Multi-Layer Perceptron (MLP), as depicted in Figure 2, though its architecture can be made more complex as needed. The importance of a token, represented as token weights, is computed as follows:

$$w_D = \text{KwNet}\left(\text{PLM}\left([\text{D}]d_1 d_2 \dots d_n\right)\right) \tag{3}$$

where $d_1 d_2 \dots d_n$ represents the token sequence of the document D, and [D] is a special token prepended to the documents as in the original ColBERT model. Instead of using ReLU, we normalize the weight of tokens by using Sigmoid as the final activation in KwNet. This provides more control over the pruning level by allowing a threshold to be set during the indexing phase. While other options such as $\tanh$ or $\arctan$ exist, Sigmoid was chosen for its favorable gradient behavior, making it easier to train.

In the ColBERT-Kw model, the similarity computation differs between training and retrieval. During training, without a predetermined token pruning threshold, the similarity between a query Q and a document D is calculated as follows:

$$sim(Q, D) = \frac{1}{l} \sum_{i \in E_Q} \max_{j \in E_D} w_D[j] \cdot E_Q[i] \cdot E_D[j]^\intercal \tag{4}$$

During retrieval, ColBERT-Kw allows us to choose a pruning threshold through a hyperparameter τ, leading to the following similarity calculation:

$$sim(Q, D) = \frac{1}{l} \sum_{i \in E_Q} \max_{j \in E_D} [\![w_D[j] \geq \tau]\!] \cdot E_Q[i] \cdot E_D[j]^\intercal \tag{5}$$

where $[\![\cdot]\!]$ is the Iverson bracket, equal to 1 if the condition is true and 0 otherwise.

Only the vectors of tokens whose importance scores meet the threshold are retained in the database during indexing. This significantly reduces the number of embeddings, leading to substantially lower computational costs in both the candidate generation and re-ranking phases of ColBERT.

This approach provides flexibility in determining a pruning threshold without requiring retraining, and it allows for directly learning the importance of tokens during training instead of relying on heuristics.

3.3 Knowledge Distillation for KwNet

Training KwNet is challenging due to the absence of explicit labels to identify key tokens within a document. The authors of ColBERTer (Hofstätter, 2022) suggest using a ReLU layer to determine whether to retain or discard tokens by regularizing token weights, encouraging sparsity. Their loss function is formulated as follows:

$$\mathcal{L} = \mathcal{L}_{sim} + \lambda \mathcal{L}_{reg} \tag{6}$$

where $\mathcal{L}_{sim}$ is computed using conventional contrastive loss functions, and $\mathcal{L}_{reg}$ is a regularization term that forces the model to assign reasonable weights to tokens, retaining only the important ones. However, our experiments show that selecting an appropriate λ is difficult, making it hard for the model to converge to the desired state.

To address the absence of labels for important tokens, we propose a knowledge distillation approach for training KwNet. This strategy leverages the MaxSim mechanism of the ColBERT model, which inherently identifies the most important tokens in documents responding to queries. These tokens are treated as keywords for ColBERT-Kw, and we train KwNet to assign high scores to them. The loss function is now defined as:

$$\mathcal{L} = \mathcal{L}_{sim} + \alpha \mathcal{L}_{distill} + \lambda \mathcal{L}_{reg} \tag{7}$$

where α controls the weight of the distillation loss $\mathcal{L}_{distill}$ in the overall objective. $\mathcal{L}_{distill}$ for a query-document pair is calculated as:

$$\mathcal{L}_{distill}(Q, D) = \sqrt{\sum_{0 \leq j \leq n} [\![j \in S]\!] \cdot \left(1 - w_D[j]\right)^2} \tag{8}$$

where S represents the set of tokens selected by MaxSim:

$$S = \left\{ \arg\max_{j \in E_D} E_Q[i] \cdot E_D[j] \right\}_{i=0}^{l} \tag{9}$$

Intuitively, minimizing $\mathcal{L}_{distill}$ equates to maximizing the weights of the tokens selected by the MaxSim mechanism of ColBERT. We also experimented with an L1 variant to evaluate its effect on KwNet's token weighting:

$$\mathcal{L}_{distill}(Q, D) = \sum_{0 \leq j \leq n} [\![j \in S]\!] \cdot |1 - w_D[j]| \tag{10}$$

Models trained using Equation 8 will be referred to as ColBERT-Kw, while those trained using Equation 10 will be referred to as ColBERT-Kw-L1. These formulations result in significantly different distributions of token weights, which will be elucidated in the upcoming experimental results section. The regularization component is straightforwardly calculated as:

$$\mathcal{L}_{reg}(D) = \frac{1}{n} \sum_{0 \leq j \leq n} w_D[j] \tag{11}$$

To prevent ColBERT-Kw from converging into a Single-Vector Bi-Encoder, we prohibit gradient backpropagation from the KwNet layer to the PLM of the document encoder. In this study, this is achieved by freezing the ColBERT component and updating only KwNet, utilizing pre-trained ColBERT parameters. Specifically, we first train an effective ColBERT model, then freeze its parameters, and subsequently train KwNet through knowledge distillation. This approach leverages the already optimized ColBERT model to facilitate more efficient learning within KwNet. By ensuring that the PLM remains unchanged, KwNet can focus on learning the importance of tokens while preserving the integrity of the pre-trained document encoder.

In scenarios where it is desirable to initialize and train ColBERT-Kw from scratch, we recommend using the $\mathrm{stopgrad}$ operator to prevent gradient backpropagation from KwNet. This approach calculates the token weights as follows:

$$w_D = \mathrm{KwNet}\left(\mathrm{sg}\left[\mathrm{PLM}\left([\mathrm{D}]d_1 d_2 \ldots d_n\right)\right]\right) \tag{12}$$

where $\mathrm{sg}[\cdot]$ indicates that gradients are not propagated back through the enclosed expression.

Other components within the formula remain unchanged. When updating the weights of ColBERT-Kw, the gradient of $\mathcal{L}$, as defined in Equation 8, can be applied directly. Alternatively, the weights of KwNet can be updated based on this gradient, while the ColBERT component is updated using a contrastive loss function, with similarity computed by the standard ColBERT formula as in Equation 2.

3.4 Leverage Domain Knowledge by Auxiliary Task Labels

Besides the keywords generated by ColBERT-Kw, other heuristics can be applied to select a good keyword set, such as domain-specific heuristics that extend beyond traditional methods based on term rarity or general grammatical rules. This strategy, tailored specifically to each dataset, significantly boosts the method's adaptability and effectiveness. In this paper, we utilize Named Entity Recognition (NER) and Part-of-Speech (POS) tagging labels to supplement additional keywords from the input document. Nouns and entities are considered as important keywords of the document. All of these keywords are included in a designated set, S, detailed further in Section 3.3. We selected NER and POS for our research due to their relevance to entity-rich datasets. Moreover, there are numerous tools available today for part-of-speech tagging and named entity recognition that provide high accuracy.

4 Experiments

In this section, we describe experiments conducted with the ColBERT and our proposed ColBERT-Kw models on a Vietnamese information retrieval dataset derived from Wikipedia. We compare the effectiveness and efficiency of both models against established baselines. Additionally, we assess the pruning performance of ColBERT-Kw relative to traditional heuristic methods, particularly its ability to minimize index size while preserving the retrieval accuracy inherent to ColBERT.

4.1 Dataset and Evaluation Metric

We evaluate the ColBERT and ColBERT-Kw models on a Vietnamese information retrieval dataset derived from the 2019 Zalo AI Challenge's Vietnamese Question Answering task[1] comprising 15,957 text documents and 5,070 unique queries from Vietnamese Wikipedia. A total of 507 queries were randomly selected for testing, with the remaining queries used for training.

We assess the performance of the models based on the following criteria:

- **Retrieval Effectiveness:** Measured using metrics such as Recall@1, Recall@10, Recall@50, and MRR@10 (Mean Reciprocal Rank at 10).

- **Retrieval Speed or Latency:** The time required to process a query and retrieve relevant documents.

[1]This specific dataset version is available in an unofficial repository: https://github.com/namnv1113/Nanibot_ZaloAIChallenge2019_VietnameseWikiQA.

- **Index Size:** Determined by the number of embedding vectors and the overall size of the index structures.

4.2 Baseline

The baseline retrieval models selected for comparison include Okapi BM25, vietnamese-bi-encoder[2] (Nguyen et al., 2024), and vietnamese-bert[3]. Okapi BM25 is a variant of the well-established BM25 algorithm, chosen for its effective term matching-based retrieval capabilities, offering a balance of accuracy, low retrieval costs, and high speed. The vietnamese-bi-encoder model, trained on the Vietnamese mMARCO dataset (Bonifacio et al., 2021), is noted for its high accuracy in text information retrieval tasks. Similarly, vietnamese-sbert demonstrates robust performance in semantic similarity tasks.

4.3 Setup

For training ColBERT and ColBERT-Kw, we used PhoBERT-base-v2 (Nguyen and Nguyen, 2020), a state-of-the-art Vietnamese language model, as the backbone PLM. The training process was divided into two phases:

- First, we trained the ColBERT model using a contrastive learning approach. Positive samples were directly sourced from the dataset, while negative samples were randomly selected from the top BM25 results, excluding the positive samples. The online contrastive loss function [4] was employed.

- After training ColBERT, we initialized ColBERT-Kw with the trained model's parameters and randomly initialized KwNet. We then optimized KwNet's parameters using the loss function from Equation 7.

The models were trained using the Adam optimizer (Kingma and Ba, 2014), with each phase conducted over 24,000 steps with a batch size of 32 and a learning rate of 3e-6. The embeddings for ColBERT and ColBERT-Kw were projected to 128 dimensions. The hyperparameters chosen for KwNet training are $\lambda = 0.1$ and $\alpha = 0.4$.

[2] https://huggingface.co/bkai-foundation-models/vietnamese-bi-encoder

[3] https://huggingface.co/keepitreal/vietnamese-sbert

[4] https://sbert.net/docs/package_reference/sentence_transformer/losses.html

We evaluated ColBERT and ColBERT-Kw in a full-ranking setup similar to ColBERT's original framework. Additionally, we trained the COIL model for comparative purposes and tested the PLAID retrieval mechanism (Santhanam et al., 2022a) with ColBERT. For heuristic-based pruning, we used a method akin to previous work (Liu et al., 2024), reducing vector counts by 50% to ensure fair comparison with ColBERT-Kw.

Training and indexing were performed on a P100 GPU provided by Kaggle, and evaluation was done on a personal computer with an Intel i7-13700H CPU.

4.4 Result

Table 1 shows the retrieval effectiveness (accuracy) of the models compared in this study. We observe that while BM25 achieves moderate accuracy on this dataset, the two single-vector bi-encoder models, although trained on extensive data, do not significantly outperform BM25. In contrast, multi-vector bi-encoder models exhibit superior performance, particularly ColBERT, which utilizes the PhoBERT backbone, achieving the highest MRR@10 among the models tested. The COIL model also surpasses the baseline models but is constrained by its matching mechanism, unable to match ColBERT's retrieval outcomes. This underscores the effectiveness of interaction between the query and document embeddings in achieving superior retrieval results.

Simpler models typically have lower accuracy but provide faster retrieval speeds. BM25 provides the fastest retrieval, followed by the single-vector bi-encoders. Among the multi-vector bi-encoder models, only COIL approaches their retrieval speed. ColBERT, due to its higher computational costs, has about three times the latency. In real-world scenarios, large datasets generate a significant number of embedding vectors, which poses larger engineering challenges. This is one reason why single-vector bi-encoders and ensemble methods are more widely used compared to multi-vector approaches.

Token pruning has proven to be an effective method for reducing ColBERT's retrieval time. Among these methods, the ColBERT-Kw models pruned based on weights threshold offer the most optimal results, reducing search time by approximately 40% - 50% while still maintaining significantly higher accuracy than conventional heuristic methods. Notably, the ColBERT-Kw model

	Recall@1	Recall@10	Recall@50	MRR@10	Latency (ms/query)
Full model					
BM25	0.3034	0.7813	0.9051	0.4628	**18.9**
vietnamese-sbert	0.2621	0.6284	0.8115	0.3744	33.0
vietnamese-bi-encoder	0.4387	0.7525	0.8743	0.5543	33.0
ColBERT	**0.5290**	**0.9479**	**0.9785**	**0.6995**	104.6
COIL	0.4548	0.8842	0.9515	0.6034	35.3
PLAID	0.5003	0.8671	0.8869	0.6483	167.1
Pruned ColBERT models					
ColBERT-First-tokens	0.4895	0.8680	0.9370	0.6261	74.1
ColBERT-Top-IDF	0.4808	0.9291	0.9740	0.6556	74.1
ColBERT-Top-Attention	0.4877	0.9022	0.9542	0.6478	74.1
ColBERT-Kw-0.7	0.5129	0.9318	0.9704	0.6802	54.9
ColBERT-Kw-L1-0.7	0.5030	0.9309	0.9740	0.6743	56.3
ColBERT-Kw-NER-0.7	0.5147	0.9372	0.9794	0.6850	66.7
ColBERT-Kw-POS-0.7	0.5218	0.9336	0.9794	0.6893	68.4
PLAID-ColBERT-Kw-0.7	0.4895	0.8573	0.8824	0.6381	87.6

Table 1: Retrieval effectiveness. ColBERT-Kw models retrieval result are reported with pruning threshold $\tau = 0.7$

	Embeddings	Size (MB)
BM25		7.1
vietnamese-bi-encoder	15957	46.7
ColBERT	811011	398.5
COIL	811011	169.6
PLAID	811523	19.8
ColBERT-First-tokens	405505	199.2
ColBERT-Top-IDF	405505	199.2
ColBERT-Top-Attention	405505	199.2
ColBERT-Kw-0.7	238209	117.2
ColBERT-Kw-L1-0.7	273603	134.6
ColBERT-Kw-NER-0.7	365156	179.6
ColBERT-Kw-POS-0.7	375952	184.9
PLAID-ColBERT-Kw-0.7	238465	6.1

Table 2: Index size

trained with POS task labels achieves the highest MRR@10 at a pruning threshold of $\tau = 0.7$ (98.5% relative to ColBERT), highlighting the efficacy of incorporating domain knowledge into the model. Thus, ColBERT-Kw and its pruning strategies emerge as an optimal solution, significantly reducing ColBERT's retrieval time with minimal trade-offs in search accuracy.

Designed for larger datasets, PLAID uses quantization and approximate calculations to reduce retrieval times and minimize index sizes for the ColBERT model. Surprisingly, on our smaller dataset, PLAID slowed down retrieval due to increased computational load during the candidate generation phase, as shown in Table 1, despite significantly reducing the index size (Table 2). When combined with ColBERT-Kw, PLAID further reduced the index size while still maintaining acceptable retrieval performance, as demonstrated in Tables 1 and 2. This suggests that integrating ColBERT-Kw with PLAID could offer substantial benefits for very large datasets, optimizing both index size and retrieval efficiency.

4.5 Ablation Study

4.5.1 Choosing Pruning Threshold τ

A key advantage of ColBERT-Kw is its ability to optimize performance with a single training session, allowing pruning thresholds (τ) to be adjusted during indexing. This flexibility surpasses methods that require a fixed pruning level during training. The choice of τ impacts both retrieval effectiveness and performance, depending on dataset size and desired balance.

Incorporating labels from auxiliary tasks increases the number of tokens identified as important, or keywords. Notably, ColBERT-Kw-POS with $\tau = 0.7$ retains fewer vectors but achieves higher accuracy than with $\tau = 0.5$. This suggests that combining a good heuristic can help the model select better keywords. And maybe pruning can eliminate noisy tokens, resulting in even

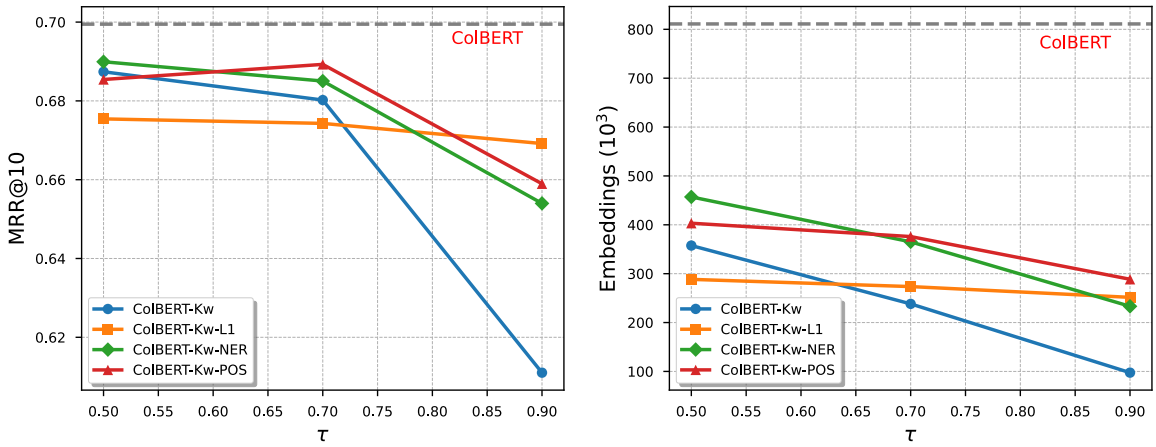

Figure 3: MRR@10 and Embeddings count for different pruning thresholds

more accurate retrievals while reducing the number of vectors used. This aspect will need further investigation in future studies.

4.6 Token Weight Distribution

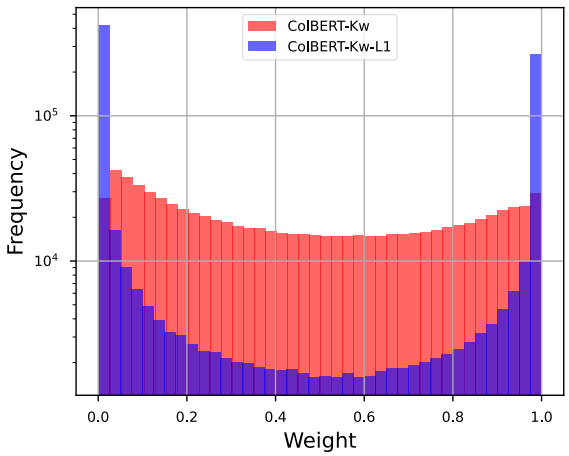

Figure 4: Token weight distribution for ColBERT-Kw and ColBERT-Kw-L1 (y-axis in logarithmic scale)

During our research, ColBERT-Kw-L1 was initially proposed. Transitioning to ColBERT-Kw by changing the distillation loss from Equation 10 to Equation 8 altered the model's token weighting behavior. Figure 4 shows that ColBERT-Kw-L1's weight distribution resembles a Bernoulli distribution, with values concentrated near 0 or 1. In contrast, ColBERT-Kw exhibits a U-shaped distribution with a broader and more even spread. Both models demonstrate varying token importance within documents. ColBERT-Kw is preferable for its flexible pruning threshold, while ColBERT-Kw-L1 is better suited for classification tasks, retaining only the most important tokens without threshold selection.

5 Conclusion

In this work, we introduced ColBERT-Kw, a model designed to facilitate token pruning for ColBERT by using KwNet to assign importance weights to tokens. With adequate training, ColBERT-Kw significantly improves ColBERT's retrieval speed and reduces storage requirements while preserving high accuracy. Our experiments on a Vietnamese Wikipedia-based dataset demonstrated that this method effectively minimizes index size and latency with minimal performance trade-offs. Our code is available at https://github.com/haihp02/Enhancing-ColBERT.

Acknowledgments

This work was supported by the School of Information and Communication Technology project, code T2024-PC-041.

References

Luiz Henrique Bonifacio, Israel Campiotti, Roberto de Alencar Lotufo, and Rodrigo Nogueira. 2021. mMARCO: A Multilingual Version of MS MARCO Passage Ranking Dataset.

Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *North American Chapter of the Association for Computational Linguistics*.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2020. Modularized Transfomer-based Ranking Framework. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4180–4190.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. COIL: Revisit Exact Lexical Match in Information Retrieval with Contextualized Inverted List. In *North American Chapter of the Association for Computational Linguistics*.

Ben Giacalone, Greg Paiement, Quinn Tucker, and Richard Zanibbi. 2024. Beneath the [MASK]: An Analysis of Structural Query Tokens in ColBERT. In *Advances in Information Retrieval: 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24–28, 2024, Proceedings, Part III*, page 431–439.

Omar Althammer Sophia Sertkan Mete Hanbury Allan Hofstätter, Sebastian Khattab. 2022. Introducing Neural Bag of Whole-Words with ColBERTer: Contextualized Late Interactions using Enhanced Reduction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, page 737–747.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring. In *International Conference on Learning Representations*.

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 the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48. ACM.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Carlos Lassance, Maroua Maachou, Joohee Park, and Stéphane Clinchant. 2021. A Study on Token Pruning for ColBERT. *Preprint*, arXiv:2112.06540.

Maroua Park Joohee Clinchant Stéphane Lassance, Carlos Maachou. 2022. Learned Token Pruning in Contextualized Late Interaction over BERT (ColBERT). In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2232–2236.

Qi Liu, Gang Guo, Jiaxin Mao, Zhicheng Dou, Ji-Rong Wen, Hao Jiang, Xinyu Zhang, and Zhao Cao. 2024. An Analysis on Matching Mechanisms and Token Pruning for Late-interaction Models. *ACM Transactions on Information Systems*, 42(5):1–28.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*, abs/1907.11692.

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*, 9:329–345.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Efficient Document Re-Ranking for Transformers by Precomputing Term Representations. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. PhoBERT: Pre-trained language models for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1037–1042.

Quang Duc Nguyen, Hai Son Le, Duc Nhan Nguyen, Dich Nhat Minh Nguyen, Thanh Huong Le, and Viet Sang Dinh. 2024. Towards Comprehensive Vietnamese Retrieval-Augmented Generation and Large Language Models. *arXiv preprint arXiv:2403.01616*.

Quang Nhat Nguyen and Huong Thanh Le. 2023. Building an efficient retriever system with limited resources. In *Advances in Information and Communication Technology*, pages 40–50. Springer Nature Switzerland.

Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage Re-ranking with BERT. *Preprint*, arXiv:1901.04085.

Duc Thang Phan and Huong Thanh Le. 2023. Utilize pre-trained phobert to compute text similarity and rerank documents for question-answering task. In *12th International Conference on Control, Automation and Information Sciences*, pages 200–205. IEEE.

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 the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847.

Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. 2022a. PLAID: An Efficient Engine for Late Interaction Retrieval. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, page 1747–1756.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022b. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021. Simple Entity-Centric Questions Challenge Dense Retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Neural Information Processing Systems*.

Shitao Xiao, Zheng Liu, Yingxia Shao, and Zhao Cao. 2022. RetroMAE: Pre-Training Retrieval-oriented Language Models Via Masked Auto-Encoder. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 538–548.