BERT-based Dense Intra-ranking and Contextualized Late Interaction via Multi-task Learning for Long Document Retrieval

Minghan Li
Univ. Grenoble Alpes, CNRS, LIG
Grenoble, France
minghan.li@univ-grenoble-alpes.fr

Eric Gaussier
Univ. Grenoble Alpes, CNRS, LIG
Grenoble, France
eric.gaussier@imag.fr

ABSTRACT

Combining query tokens and document tokens and inputting them to pre-trained transformer models like BERT, an approach known as interaction-based, has shown state-of-the-art effectiveness for information retrieval. However, the computational complexity of this approach is high due to the online self-attention computation. In contrast, dense retrieval methods in representation-based approaches are known to be efficient, however less effective. A tradeoff between the two is reached with late interaction methods like ColBERT, which attempt to benefit from both approaches: contextualized token embeddings can be pre-calculated over BERT for fine-grained effective interaction while preserving efficiency. However, despite its success in passage retrieval, it's not straightforward to use this approach for long document retrieval. In this paper, we propose a cascaded late interaction approach using a single model for long document retrieval. Fast intra-ranking by dot product is used to select relevant passages, then fine-grained interaction of pre-stored token embeddings is used to generate passage scores which are aggregated to the final document score. Multi-task learning is used to train a BERT model to optimize both a dot product and a fine-grained interaction loss functions. Our experiments reveal that the proposed approach obtains near state-of-the-art level effectiveness while being efficient on such collections as TREC 2019.

CCS CONCEPTS

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

KEYWORDS

Neural IR, BERT-based models, late interaction, multi-task learning

ACM Reference Format:
Minghan Li and Eric Gaussier. 2022. BERT-based Dense Intra-ranking and Contextualized Late Interaction via Multi-task Learning for Long Document Retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3477495.3531856

1 INTRODUCTION

Information retrieval (IR) plays an important role in our daily life in the era of big data. Retrieving relevant documents given a query is the central part of many applications in our daily life, e.g., web search. Deep neural networks have shown great success on a variety of tasks including information retrieval [3, 5, 12, 22, 27] with pre-trained Transformer [26] architectures like BERT [4] leading to state-of-the-art performances [2, 15–17, 20, 22]. BERT-based neural IR approaches can be classified into three categories [15]: interaction-based methods, representation-based methods and late-interaction methods. This first method , like a vanilla BERT model [22], where the query tokens and document tokens are concatenated as BERT inputs and applied full self-attention, is viewed to be extremely effective [7] but suffers from high computational complexity. On the other hand, representation-based methods generate two representations [24] for a query and a document respectively. When the document representations can be pre-stored, this method enables efficient fast retrieval at the expense effectiveness. To take advantage of both approaches, late-interaction methods have been proposed, ColBERT [15] being certainly the most well-known representative of this category. In ColBERT, token level passage embeddings are pre-stored, which are then late interacted with query embeddings to produce a relevance score. This method is slightly more efficient than representation-based methods, but definitely more effective.

ColBERT was primarily used for passage ranking and, as most BERT methods, suffers from the major drawback that it cannot directly handle long documents. Although researchers [9, 16–18] has proposed some methods for long document information retrieval, they are designed for interaction-based methods that are computational expensive. So far, there have been no attempts to adapt late interaction methods to long documents.

We address this problem here through a BERT-based dense intra-ranking and contextualized late interaction (ICLI) with multi-task learning. Efficiency is guaranteed by the pre-calculation of self attention, and effectiveness by the fact that pre-stored token embeddings are interacted in a fine-grained way. To the best of our knowledge, this is the first attempt to adapt a late interaction method for long document retrieval. Experimental results show that the proposed approach obtains near SOTA level effectiveness while being efficient on such collections as TREC 2019.

2 RELATED WORK

As already mentioned, neural IR can be classified into three types: interaction-based, representation-based [6] and late interaction-based methods. The first effective interaction-based neural IR approaches, as DRMM [5], KNRM [27] or Conv-KNRM [3], were proposed before the introduction of BERT. After transformer-based BERT models were proposed, the field of neural IR has seen rapid improvements. Nogueira and Cho [22] proposed a simple usage of BERT for passage re-ranking where the query and passage tokens are concatenated and processed by the BERT and the [CLS] output of BERT is used to assess the relevance, and got results that outperformed other neural IR models largely. Hofstätter et al. [11] introduced the TK model which relies on transformers to learn contextualized embeddings and kernel matching for ranking. Early representation-based models as DSSM [12] were appealing because of their extremely low latency. These models have also benefited from BERT-based architectures, leading to so-called dense retrieval models [24, 28]. Despite their efficiency, representation-based models are however less effective than interaction-based models. A trade-off between the two approaches is realized with late interaction methods like ColBERT [15], which relies first on separate representations for queries and documents and which approximates the effectiveness of interaction-based methods in a late interaction step. It places the high latency passage processing by BERT to offline stage and the contextualized token embeddings are stored, which enable fine-grained late interaction. ColBERT is slightly less efficient than dense retrieval methods, but more effective. However, as other BERT-based models, this model cannot directly handle long documents, due to the complexity of the self-attention step.

To deal with long document retrieval, a variety of methods have been proposed, e.g., modified attention methods like Longformer and QDS-Transformer [1, 13], sliding window and local relevance aggregation like TKL [10], passage representation aggregation methods like PARADE [16], and the recent selecting key block/passage for evaluation methods like KeyBLD and IDCM [9, 17, 18]. Recently, the KeyBLD model family [17, 18], that first filters a long document by selecting key blocks on which to ground the document relevance, has shown SOTA level performance while also being memory efficient. In parallel, the IDCM model [9] was proposed, the core idea of which is also to first select key passages on which to ground the document relevance. In this paper, we introduce a late interaction method for long document retrieval based on the same idea but in a different way that enables both effectiveness and efficiency. The details are described in Section 3.

3 METHOD

As mentioned above, the selecting key block related methods [9, 17, 18] have been shown to achieve SOTA level effectiveness for long document information retrieval. Nevertheless, these models have been designed for interaction-based models which have high computational complexity due to their online self-attention computation and not good solutions for real world online search scenarios where low latency is crucial. We propose to extend them to late interaction retrieval methods for long documents by using a cascaded ranking approach based on dense intra-ranking and late interaction. However, designing this is non-trival for being both effective and efficient. In the following, we firstly introduce the overall architecture and then describe the details and the reason of some choices.

The overall architecture of the proposed method is depicted in Fig. 1 and described later.

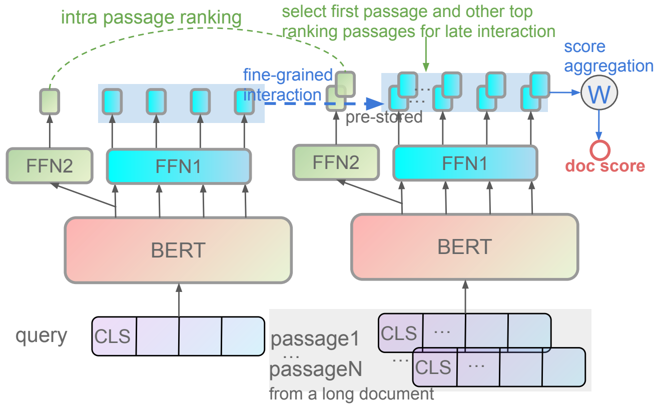

Figure 1: The architecture of proposed approach. Contextualized embeddings are calculated by the BERT model and a feedforward neural network for late interaction. The [CLS] embedding is also inputted to another feedforward neural network for passage ranking. The selected passages' late interaction scores are aggregated to obtain the document relevance score. The document tokens can be pre-stored.

3.1 Contextualized Document Embedding

The ColBERT model's efficiency comes from its offline pre-computed contextualized passage token embeddings. Here, we want to also rely on the pre-stored contextualized document embeddings to enable fast retrieval. Due to the quadratic complexity of self-attention mechanism, transformer based models including BERT can only handle limited number of tokens, so they are not able to process long documents directly. To tackle this, following previous work [9, 16, 17], we firstly segment a long document into passages, which can be inputted to a BERT model separately to obtain contextualized embeddings. During training, the BERT model can be learned end-to-end and enables storing contextualized embeddings.

Given a document D, we segment it into passages $P_1P_2...P_k$, which can be done in a sliding window way. For each passage, with a BERT tokenizer, we can obatin its tokens $p_1p_2...p_m$, then BERT's [CLS] token is prepended to the tokens of each passage. The BERT model will compute the contextualized token embeddings $E_p^{cls}E_p^1...E_p^m$, where each E_p is of dimension 768 ($dim(E_p = 768)$) for a bert-base-uncased model (as shown in the right part of Fig. 1).

Each embedding E_p is then passed into a one-layer feedforward neural network FFN_1, referred to as *compressor1* (the blue module in Fig. 1), to obtain a low dimensional vector for late interaction: $V_p = FFN_1(E_p)$, where $dim(V_p) = 128$. It is worth mentioning that the [CLS] embedding is also passed to a one-layer feedforward neural network FFN_2, referred to as *compressor2* (the green module in Fig. 1), for dense intra-ranking of the passages in a document. This is to say, $V_p^{cls_1} = FFN_1(E_p^{cls})$, $V_p^{cls_2} = FFN_2(E_p^{cls})$ with, for each V_p^{cls}, $dim(V_p^{cls}) = 128$. The choice of using two compressors is based on the fact that a vector for intra-ranking should contain query/passage representation information while a vector participating into late interaction should be trained together with other tokens in a different way. Using two compressed vectors allows one to capture these differences and gives more flexibility.

Online computation is used during training. During deployment, the contextualized tokens of each document are pre-computed and stored for efficient late interaction.

3.2 Contextualized Query Embedding

Similar to a passage in a document, BERT's [CLS] token is prepended to the tokenized query tokens, which are passed to the BERT model to obtain contextualized query token embeddings $E_q^{cls}E_q^1...E_q^n$, with, for each E_q, $dim(E_q) = 768$. Then using FFN_1, one obtains a low dimensional vector V_q for each E_q. For the [CLS] token, $V_q^{cls_1}$ and $V_q^{cls_2}$ are also calculated.

Different from documents, the query is always computed online, which is also the same case for ColBERT and dense retrieval models. Nevertheless, the computational cost is relatively small as the queries are shorter and only required to be computed once to retrieve different documents.

3.3 Intra-ranking for Key Passage Filtering

The token embeddings of a long document can be pre-stored, however, they are normally too long which may result in high latency and may contain noisy information. Previous works [9, 18] first select key passages according to the query to make it more efficient and even more accurate. BM25 and learning based methods can be used where the later normally performs better as it enables semantic matching. In the case of late interaction, we also want to use this step and to use learning based method for informative passage selection. To rely on pre-stored embeddings, inspired by dense retrieval where a query and a passage are represented by a low dimensional vector respectively, normally using the [CLS] embedding, we want to rely on this which allows us to do semantic matching and is naturally part of the BERT model. To do so, the [CLS] embedding of a passage or a query from BERT is inputted to a compresser layer2 (the FFN_2 module in Fig. 1), to obtain the representation of a passage. Dot product is used during inference, to select the most informative chunks, with the pre-stored passage representations and the online query representation.

As the first passage of a document usually carries important information, as illustrated in [18], we always select this passage in our approach. This strategy is also consistent with truncation-based methods which truncate long documents and rely only on their beginning in order to apply BERT-based models. Given the representation $V_q^{cls_2}$ of a given query and pre-stored representations $V_p^{cls_2}$ of passages in a given document, we use the standard dot product to score passages:

$$S_{q,p}^1 = V_q^{cls_2} \cdot (V_p^{cls_2})^T. \tag{1}$$

This selection process provides "top"-k passages for each query-document pair. Having the first passage in the "top"-k list furthermore allows one to train compressor2 through a standard ranking loss, as described below. During training, this step is eval model for PyTorch, which means no backpropagation.

The above approach finally amounts to learning a representation useful for selecting passages and is in line with previous work [18] that has shown that learning how to select key blocks in a document can outperform methods that select key blocks using standard ranking functions as BM25 [25].

3.4 Fine-grained Late Interaction

After having selected the k passages, we adopt, for each passage, the late interaction approach of ColBERT to obtain the query-passage relevance score:

$$S_{q,p}^2 = \sum_{i \in [cls_1, 1...n]} \max_{j \in [cls_1, 1...m]} V_q^i \cdot (V_p^j)^T. \tag{2}$$

These scores are then simply aggregated through a weighted sum:

$$S_{q,d} = w_1 S_{q,p_1}^2 + ... + w_k S_{q,p_k}^2, \tag{3}$$

where the weights $\{w_1, ..., w_k\}$ are real numbers.

3.5 Multi-task Learning

Since the [CLS] contextualized vectors are used for two tasks: intra passage ranking (Section 3.3) and late interaction (Section 3.4), we adopt a multi-task learning approach to ensure that both tasks are taken into account to fine-tune the BERT model. As mentioned before, (a) the first passage of a document usually contains relevant information and is always selected, (b) this first passage is furthermore always used by relatively strong baselines based on truncation, and (c) its use ensures the training of the first task. Indeed, we train the [CLS] vector to be used for intra-passage ranking using the first passage of documents as explained below. The second task directly relies on the scores of the selected passages (Eq. 2) and the document labels. The loss functions associated with these tow tasks are given in the following section.

3.6 Loss Functions

Following [9], we use the pairwise RankNet loss for each task, defined by:

$$\mathcal{L}(q, d^+, d^-; \Theta) = -\log(Sigmoid(S_{q,d^+} - S_{q,d^-})),$$

where q is a query, (d_q^+, d_q^-) is a positive and negative training document pair for q, Θ represents the parameters of the model and $S_{q,d}$ is the score provided by the model for document d. For task 1, the loss function is given by:

$$\mathcal{L}_1(q, d^+, d^-; \Theta) = RankNet(q, S_{q,p_{1+}}^1, S_{q,p_{1-}}^1),$$

where $S_{q,p_{1+}}^1$ (resp. $S_{q,p_{1-}}^1$) is the relevance score of the first passage of a positive (resp. negative) document for query q according to Eq. 1. For task 2, the loss is given by:

$$\mathcal{L}_2(q, d^+, d^-; \Theta) = RankNet(q, S_{q,d_+}, S_{q,d_-}),$$

where S_{q,d_+} (resp. S_{q,d_-}) is the relevance score of a positive (resp. negative) document for query q according to Eq. 3.

As the two losses may have different scales, we combine them to obtain the final loss $\mathcal{L}(q, d^+, d^-; \Theta)$ following [19], which adjusts the proposal of [14] to enforce positive regularization:

$$\mathcal{L} = \frac{1}{2\sigma_1^2}\mathcal{L}_1 + \frac{1}{2\sigma_2^2}\mathcal{L}_2 + \log(1 + \sigma_1^2) + \log(1 + \sigma_2^2),$$

where σ_1 and σ_2 are parameters of the model.

4 EXPERIMENTS

In this section, we evaluate the proposed approach for both effectiveness and efficiency. Besides, an ablation study is performed to analyze the model.

4.1 Datasets

We make use here of the widely used test collection TREC 2019 Deep Learning Track Document Collection. We use two test sets: TREC 2019 and 2020 test queries, where the task is to rerank top 100 documents for each test query. The first test set is widely studied [9, 13, 18] on MS MARCO v1 corpus. We here also evaluate it on MS MARCO v2 corpus[1], which is said to be larger, cleaner and more realistic. To be specific, for TREC 2019 and 2020 test set, we use the same official data with MS MARCO v1 corpus, where the training set is 'msmarco-doctrain-top100', validation set is ' msmarco-docdev-top100'. For TREC 2019 test set, we also additionally evaluate it on the MS MARCO v2 corpus, where the training set is 'docv2_train_top100', validation set is 'docv2_dev_top100'.

4.2 Baseline Models

The proposed approach is compared with 5 baselines:

- **BM25**: we use the BM25 [25] implementation of Anserini [29], with default hyperparameters;
- **BERT-CAT**: this is a BERT interaction-based [22] baseline;
- **TK**: this model [11] generates contextualized embeddings using Transformer and does kernel matching.
- **TKL**: this model [10] extends TK model for long documents;
- **ColBERT**: this is the SOTA late interaction model [15] for passage ranking.

In addition, we also propose to use BM25 to do intra-ranking to select passages. The other steps are unchanged but we do not need multi-task learning in this case as there is no learning of the selection module. Only late interaction loss, $\mathcal{L}_2$ is required. In this setting, we use the BM25 model proposed in [17] which has provided very good results for block selection on several collections.

4.3 Experimental Settings

Our implementation is based on the matchmaker open-source framework[2], using automatic mixed precision [21]. For the MS MARCO v1 corpus, the goal is to rerank the official top 100 documents. For the MS MARCO v2 corpus, as TREC 2019 DL has no official top retrieved documents, we rely on Anserini [29] with default hyperparameters to obtain top 100 documents which are then used for reranking. All neural-IR models, except TKL, are trained for 40000 steps where each step contains 8 positive-negative pairs. For TKL, each step is set to 32 pairs and is trained for 10000 steps. All models are trainied using the pairwise RankNet loss. The negative documents in the positive-negative pairs are sampled randomly, from each query's official top 100 documents that have no label. We conduct 10 validations during training to store the best performing model.

For BERT-based models, the learning rate for BERT is set to 1.0e-05, the other learning rates are set to 1.0e-3. TK and TKL use 1.0e-4, the learning rate for kernels is 1.0e-3. Adam optimizer is used to train the models. For BERT-CAT, TK and ColBERT, we only consider the first 400 tokens; indeed, as these tokens are concatenated with the query, this ensures that one does not exceed the maximum allowed size of 512 tokens in BERT (note that previous studies, as [10], only considered 200 tokens for these kind models). The BERT model used for queries and documents is shared. The maximum document length is set to 3000 tokens, while the passage length is set to 200 tokens (without overlap as we did not see significant difference during preliminary experiments). This means that a document can split into a maximum of 30 passages (when the document length is shorter than 3000, the obtained passages can be less). For TKL, we set the passage size to 40, with an overlap of 10, as done in the original paper. Both TK and TKL use lowercased texts as GloVe [23] embeddings are lowercased version, while BERT based models use original texts (bert-base-uncased model is used and in this case every token will be made lowercase automatically). For the intra-ranking step, we select 4 passages (an extreme case is the original passage number is less than 4, if so we pad 0 as the scores).

To learn both the complete BERT model and the aggregation scores (Eq. 3), we first fix the aggregation scores to [0.4, 0.3, 0.2, 0.1] and learn BERT, prior to fix BERT parameters and adjust the aggregation scores. The choice for the initial values for the aggregation scores is arbitrary and simply reflects the fact that passages with higher relevance scores are more important for deciding the relevance of the entire document.

Finally, results are reported according to NDCG@10 and MAP, two standard metrics on the collections considered.

Table 1: Results on *TREC 2019 DL* collection of MS MARCO v1 and v2 corpus. Best results are in bold.

	MS MARCO v1		MS MARCO v2	
Model	NDCG@10	MAP	NDCG@10	MAP
BM25	0.5176	0.2434	0.2368	0.0865
BERT-CAT	0.6519	0.2627	0.3754	0.1144
TK	0.5850	0.2491	0.3290	0.1086
TKL	0.6213	0.2656	0.3351	0.1108
ColBERT	0.6504	0.2688	0.3788	0.1133
ICLI-BM25	0.6806	0.2703	0.3926	**0.1160**
ICLI-dot	**0.7048**	**0.2768**	**0.4049**	0.1146

Table 2: Results on *TREC 2020 DL* dataset, corpus MS MARCO v1. Best results are in bold.

Model	NDCG@10	MAP
BM25	0.5286	0.3793
BERT-CAT	**0.6211**	**0.4112**
TK	0.5732	0.3660
TKL	0.5677	0.3633
ColBERT	0.5951	0.3907
ICLI-BM25	0.5940	0.3783
ICLI-dot	0.6042	0.3938

4.4 Results

The results obtained on the TREC 2019 DL test collection with MS MARCO v1 and v2 are shown in Table 1. With MS MARCO v1,

[1]https://microsoft.github.io/msmarco/TREC-Deep-Learning-2021
[2]https://github.com/sebastian-hofstaetter/matchmaker

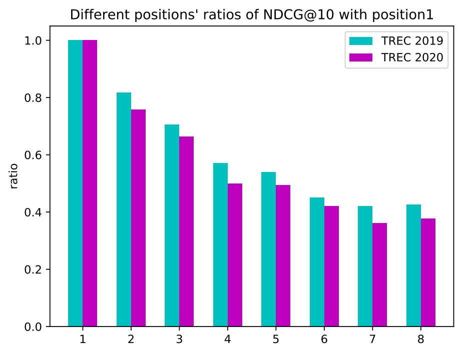

Figure 2: The NDCG@10 result of different positions compared with the first position.

the proposed approach ICLI with dot product for passage filtering achieves SOTA results: for NDCG@10, it reaches 0.7048, which is 8.36% higer than ColBERT. With MS MARCO v2, it also obtains the best result on NDCG@10. Using BM25 to filter passages for late interaction, ICLI-BM25 is the second best method on MS MARCO v1, the best method on MAP and second best on NDCG@10 on MS MARCO v2. This proves that the proposed approach is effective compared to other late interaction methods.

The results obtained on the TREC 2020 DL test collection are displayed in Table 2. As once can see, BERT-CAT using only the first 400 tokens is the best performing method, while the proposed approach is only slightly better than ColBERT. A similar trend is observed on the TK and TKL models as TK, with only the first 400 tokens, obtains better results than TKL with 3000 tokens. These results are consistent with the findings reported in [8]: the authors also observed that for TKL, the 2K model outperforms the 4K model on TREC 2020.

To better understand what’s happening on the TREC 2020 test collection, we display in Fig. 2, for both TREC 2019 and TREC 2020, the ratio of the NDCG@10 score of the i^{th} (from 1 to 8) passage of a document. To be specific, we use each passage’s relevance score as the document relevance score to calculate the NDCG@10 with the label. Each passage’s relevance score with the query is obtained using Sentence-BERT [24] pre-trained on MS MARCO passage collection[3]. We consider here the first 8 passages, each passage containing 400 tokens. As one can note, the relevance information decreases with the position, and, compared to passages in TREC 2019, passages in TREC 2020 tend to be less relevant when their position in the document increases. We believe that this, at least partly, explains the unexpected results observed here and in previous studies on TREC 2020. This said, the ICLI-dot model is still comparable with ColBERT and better than the TK family on this collection.

4.5 Reranking Latency

Following [15], latency is used to measure the average time (in seconds) for loading the pre-stored document vectors from disk to the GPU, for computing the query representation and for applying the fine-grained interaction module to rerank 100 documents for a query on the TREC 2019 MS MARCO v1 test collection (total 43 queries). Average latency results for reranking, on a RTX 8000 GPU, each query’s top 100 documents for TREC 2019 test set are reported in Table 3. The average latency for BERT-CAT amounts to 1.0234, to 0.5002 for TKL, and to 0.3420 for ICLI-dot. This demonstrates that the proposed approach is more efficient than BERT-CAT, by a factor of 3, and TKL, by a factor of 1.46.

[3]The msmarco-distilbert-base-v4 version from https://www.sbert.net/docs/pretrained-models/msmarco-v3.html

Table 3: Average reranking latencies (seconds) on *TREC 2019 DL* test set, corpus MS MARCO v1 for 100 documents with a query. Best result is in bold.

Model	Latency
BERT-CAT	1.0234
TKL	0.5002
ICLI-dot	**0.3420**

Table 4: Ablation study on *TREC 2019 DL* dataset, corpus MS MARCO v1.

Model	NDCG@10	MAP
Complete	0.7048	0.2768
W/o compressor1	0.6798	0.2687
Compressor2 = compressor1	0.6975	0.2719
Score aggregation: equal weights	0.6634	0.2614
Score aggregation: learned weights	0.6646	0.2725

4.6 Ablation Study

Lastly, Table 4 reports the results of the ablation study we performed on the ICLI-dot model on MS MARCO v1. We provide in the first line the results obtained with the complete model. The second line corresponds to removing the compressor for intra-ranking module, in which case the model uses the original 768 dimensional [CLS] embeddings (this embedding is also directly learned with task 1). The third line means without compressor2, in which case the model uses the same compressor for intra-ranking and late interaction. The fourth line corresponds to initializing the aggregation weights (Eq. 3) to $[0.25, 0.25, 0.25, 0.25]$. Lastly, the fifth line shows the results when directly learning the score aggregation weights. As one can note, comparing with the proposed design, the above modifications show lower results, suggesting the importance of using different compressors and validating our choice of learning weights for score aggregation by partly decoupling their learning from the one of the BERT model.

5 CONCLUSION

In this paper, we have attempted to adapt late interaction methods for long document retrieval by first learning the [CLS] vectors for fast intra-passage ranking, and then by applying late interaction on contextualized token vectors to obtain the fine-grained relevance scores for each selected passage. Score aggregation and multi-task learning methods are furthermore used to combine the various ingredients of our approach. Experimental results demonstrate the efficiency and efficacy of the proposed approach on such collections as TREC 2019.

ACKNOWLEDGMENTS

This work has been partially supported by MIAI@Grenoble Alpes (ANR-19-P3IA-0003) and the Chinese Scholarship Council (CSC) grant No.201906960018.

REFERENCES

[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[2] 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*. 985–988.

[3] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 126–134.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018).

[5] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM international on conference on information and knowledge management*. 55–64.

[6] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. 2020. A deep look into neural ranking models for information retrieval. *Information Processing & Management* 57, 6 (2020), 102067.

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

[8] Sebastian Hofstätter and Allan Hanbury. 2020. Evaluating Transformer-Kernel Models at TREC Deep Learning 2020. In *TREC*.

[9] Sebastian Hofstätter, Bhaskar Mitra, Hamed Zamani, Nick Craswell, and Allan Hanbury. 2021. Intra-document cascading: Learning to select passages for neural document ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1349–1358.

[10] Sebastian Hofstätter, Hamed Zamani, Bhaskar Mitra, Nick Craswell, and Allan Hanbury. 2020. Local Self-Attention over Long Text for Efficient Document Retrieval. In *Proc. of SIGIR*.

[11] Sebastian Hofstätter, Markus Zlabinger, and Allan Hanbury. 2020. Interpretable & Time-Budget-Constrained Contextualization for Re-Ranking. In *ECAI 2020*. IOS Press, 513–520.

[12] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2333–2338.

[13] Jyun-Yu Jiang, Chenyan Xiong, Chia-Jung Lee, and Wei Wang. 2020. Long Document Ranking with Query-Directed Sparse Transformer. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. 4594–4605.

[14] Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7482–7491.

[15] 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*. 39–48.

[16] Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2020. PARADE: Passage representation aggregation for document reranking. *arXiv preprint arXiv:2008.09093* (2020).

[17] Minghan Li and Eric Gaussier. 2021. KeyBLD: Selecting Key Blocks with Local Pre-ranking for Long Document Information Retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2207–2211.

[18] Minghan Li, Diana Nicoleta Popa, Johan Chagnon, Yagmur Gizem Cinar, and Eric Gaussier. 2021. The Power of Selecting Key Blocks with Local Pre-ranking for Long Document Information Retrieval. *arXiv preprint arXiv:2111.09852* (2021).

[19] Lukas Liebel and Marco Körner. 2018. Auxiliary tasks in multi-task learning. *arXiv preprint arXiv:1805.06334* (2018).

[20] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1101–1104.

[21] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2018. Mixed Precision Training. In *International Conference on Learning Representations*.

[22] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[23] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[24] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3982–3992.

[25] Stephen E. Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (2009), 333–389.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 6000–6010.

[27] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*. 55–64.

[28] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*.

[29] Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using Lucene. *Journal of Data and Information Quality (JDIQ)* 10, 4 (2018), 1–20.