IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 13, 2022, accepted April 30, 2022, date of publication May 23, 2022, date of current version May 26, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3176894

Multitask Fine-Tuning for Passage Re-Ranking Using BM25 and Pseudo Relevance Feedback

MEOUNGJUN KIM[1] AND YOUNGJOONG KO[2]
[1]Department of Artificial Intelligence, Sungkyunkwan University, Suwon 16419, South Korea
[2]Department of Computer Science and Engineering, Sungkyunkwan University, Suwon 16419, South Korea

Corresponding author: Youngjoong Ko (youngjoong.ko@gmail.com; yjko@skku.edu)

This work was supported in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant funded by the Korea Government (MSIT) (A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques) under Grant 2020-0-00368, in part by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government (MSIT) under Grant NRF-2020R1A2C2100362, and in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant funded by the Korea Government (MSIT) (AI Graduate School Support Program, Sungkyunkwan University) under Grant 2019-0-00421.

ABSTRACT Passage re-ranking is a machine learning task that estimates relevance scores between a given query and candidate passages. Keyword features based on the lexical similarities between queries and passages have been traditionally used for the passage re-ranking models. However, such approaches have a limitation; it is difficult to find semantic and contextual features beyond word-matching information. Recently, several studies based on neural pre-trained language models such as BERT overcome the limitations of traditional keyword-based models and they show significant performance improvements. Such ranking models have the advantage of finding the contextual features of queries and documents better than traditional keyword-based methods. However, these deep learning-based models require large amounts of data for training. Such training data is usually manually labeled with high cost, and how to utilize the data efficiently is an important issue. This paper proposes a fine-tuning method for efficient training of the neural re-ranking model. The proposed model utilizes data augmentation by simultaneously learning the ranking and MLM tasks during the fine-tuning stages. For the MLM task, different parts of a passage are masked at each training epoch. Even if only one pair of query and passage is given, the model is exposed to diverse cases with passages dynamically masked from the one. Also, the probability distribution of term importance is trained on the model. We calculate term importance weight by two novel methods using BM25 and pseudo relevance feedback. Terms are sampled and masked according to the importance weight. The ranking model learns representation based on the term weight distribution by executing the MLM task. A novel method with pseudo relevance feedback is applied for calculating term importance. It enables the neural ranking models to form representation according to feedbacks from an initial search stage. The proposed model is trained with data from the MS MARCO leaderboard for the re-ranking task. Our model achieves the state-of-the-art MRR@10 score in the leaderboard except for the ensemble-based method. In addition, our model demonstrates significant performance in three different evaluation metrics: MRR@10, Mean Rank, and Hit@(5,10,20,50).

INDEX TERMS Information retrieval, passage ranking, pre-trained language model, self-supervised learning.

I. INTRODUCTION

Passage re-ranking task is one of the important pipelines of the information retrieval system. When a user query is entered into the search engine, two ranking stages are executed for finding relevant passages: the full ranking stage and the re-ranking stage. The candidate passages are searched from a pre-built index in the full ranking stage according to lexical similarity with the query. In the re-ranking stage, the system extracts features from the query and the candidate passages. A machine learning-based ranking model estimates the relevance between the query and the passages using the features. After that, a limited number of passages with high relevance scores are exposed to the user. Machine learning

The associate editor coordinating the review of this manuscript and approving it for publication was Yizhang Jiang.

methods for the re-ranking task include SVM [1], neural network [2], [3], and gradient boosted decision trees [4]. Recently, neural network models with word embedding are replacing the feature-based models in the re-ranking task [5], [6]. Especially, rankers based on pre-trained language models demonstrate significant performance improvements [7]–[9]. Such approaches utilize semantic and contextual information between the query and passages while not depending on word matching features.

Nevertheless, large amounts of data are needed to train the pre-trained models for passage re-ranking. In reality, the relevance between a query and a passage in information retrieval needs to be manually labeled, which results in a costly process. We propose a new fine-tuning method based on a masked language model (MLM) that is typically used in pre-trained language models. The proposed multitask-learning-based fine-tuning method for the ranking model jointly trains both the relevance prediction and MLM tasks. We employ MLM task as a self-supervised learning approach for passage re-ranking without the need for additional training data; it is usually trained with a general and large corpus for pre-training language models [10], [11]. However, there are recent reports on some research cases for training MLM tasks on pre-trained models to enhance language understanding regarding specific domains or tasks [12]. Our multitask-based fine-tuning method is different from these methods in that it does not need much training time for the additional training task. Besides, we show that MLM training in the fine-tuning stage enables data augmentation empirically.

One of the recent changes in the re-ranking models is that word embedding methods replace handcrafted features for model input [5], [6], [9]. There is an advantage that the models can learn deep representation beyond term exact matching information. However, those methods take the query and the passage as a concatenated sequence without consideration of the relation between them. This paper suggests methods that trains the information related to the ranking task while sustaining the representation ability of deep learning-based word embedding. We devised two novel methods that calculate the importance weight for terms in the passage. The probability distribution of the weights is trained on the model. For the MLM task during the fine-tuning stage, the proposed method masks terms with masking probabilities based on the importance weights. It is different from previous studies with pre-trained language models that mask all the terms with equal probability [10], [11]. For calculating the term importance scores, we refer to two statistical models frequently used in information retrieval: BM25 and pseudo relevance feedback. BM25 is a Poisson distribution-based model representing lexical similarities between the query and the passages [13]. Pseudo relevance feedback is one of the methods for query expansion utilizing feedbacks from the users about previous ranking results [14]. In information retrieval, above two methods are usually used independently with each other. However, this work connects the two methods by constructing the initial ranking for pseudo relevance feedback using scores from BM25.

Our model obtained a state-of-the-art MRR@10 score on MS MARCO passage re-ranking task leaderboard [15] except for one ensemble-based model. Besides, the proposed model demonstrated a 3.4%p higher MRR@10 score and 8.3 higher mean rank than the baseline. The model achieved about 5%p higher scores at Hit@(5,10,20,50). It was a significant performance in all three metrics with different evaluation standards. In addition to the significant performance, our model has another advantage of efficient processing time in both training and testing stages even though a self-supervised approach is used.

This paper is an expanded version of our previous work [16] about neural re-ranking models. In this study, we calculated term importance scores using pseudo relevance feedback as well as BM25. Our model was evaluated by MRR@10, Mean Rank, and Hit@(5,10,20,50) and it achieved significant improvements. Expanded contributions in this version are as follows:

- Term importance scores are calculated using the method based on pseudo relevance feedback. It is an especially adaptive approach to information retrieval tasks where user feedback is important.
- Through additional experiments, we prove that our multitask fine-tuning is more effective than existing approaches when the model needs to understand specific domains or tasks.
- Our model with the new method achieved significant performance in diverse metrics with different evaluation standards.

II. RELATED WORK

A. PASSAGE RE-RANKING MODELS

Passage re-ranking involves finding passages that are relevant to a given query using the candidates from an initial ranking stage. The aim of re-ranking is to achieve balance between the ranking performance and efficiency. Traditional re-ranking models mainly depended on machine learning-based methods, also known as ''learning to rank.'' Such methods utilized handcrafted features devised by a human. Ranking SVM [1] utilized a support vector machine model for solving ordinal regression problems. In the problem, it was needed to construct a ranking of the labels according to their values. RankNet [2] was a ranking model based on a two-layer neural network. The neural network architecture was also utilized in LambdaRank [3]. The model was trained by implicit loss function because it was hard to directly optimize the ranking models according to the evaluation metrics. LambdaMart [4] computed an optimal combination of multiple ranking models based on LambdaRank and gradient boosted decision trees.

Although deep learning methods reduced the need for feature engineering, several works tried to leverage useful information about query–passage relevance while employing the powerful learning ability of the neural networks.

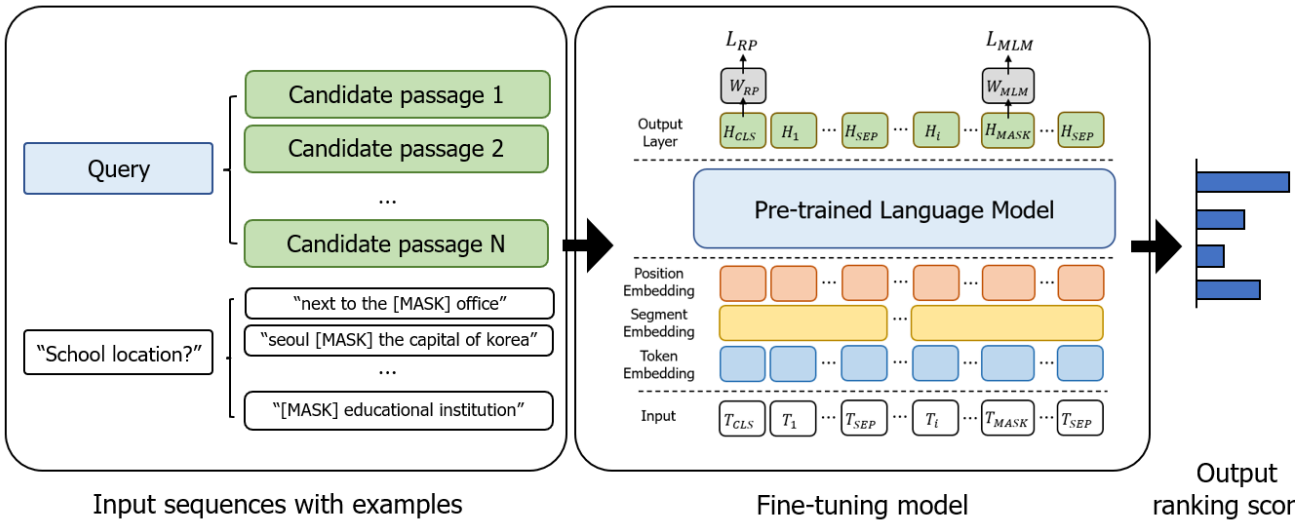


FIGURE 1. Architecture of the proposed model.

DUET [5] was a hybrid neural ranking model including layers for learning query–passage exact matching and other layers for semantic matching. The term interaction matrix between a query and a passage was entered into exact matching layers and term embeddings of input sequence into semantic matching layers. KNRM [6] also used query–passage matching features. In this model, a matrix of cosine similarity scores between query embedding and passage embedding was computed. The similarity matrix was converted to soft match features and it was entered into a linear layer for calculating ranking scores.

B. PRE-TRAINED LANGUAGE MODELS

In natural language processing, the goal of pre-trained language models [10], [11], [17] is learning distributed representations of words. For this purpose, two stages of the training process are executed: pre-training and fine-tuning. In the pre-training stage, the models learn general language representation from the massive corpus using self-supervised methods like masked language model (MLM) and next sentence prediction (NSP). In the fine-tuning stage, parameters of the pre-trained models are updated again for a specific target task. It is a transfer learning-based approach to enhance the ability for a downstream task while leveraging language information gained from pre-training. Recently, passage re-rankers based on the pre-trained language models have demonstrated excellent performances. Nogueira *et al.* reported the first successful study based on application of BERT to re-ranking tasks [8]. Sun *et al.* trained a BERT-based multistage ranking model with a target passage corpus and MLM [9]. This work was different from our proposed model in that additional pre-training steps with the MLM and NSP on the training corpus were needed. Boualili *et al.* added new marking tokens to the input sequences in the BERT model to check for exactly matched words between a query and a passage [18]. This method was similar to the proposed one because it enabled the model to learn about the input tokens that are more important for ranking. However, our model estimates the term importance in a passage using the probability model instead of the exact matching features. We utilize BM25, the probability model widely used in information retrieval. The probability that a specific term appear at the passage is normalized and used for the importance score. More detailed explanation about BM25 including a formula is available later in this chapter.

A cross-encoder method and a dual-encoder method are pre-trained language model architectures for ranking. In the cross-encoder method, a query and a passage are concatenated as an input sequence. Full cross attention is computed between token representations for the query and the passage. This approach accomplishes powerful ranking performance in general even though it depends on costly full attention operation. In the dual-encoder method, the query and the passage are embedded separately, and the similarity between the two embedding vectors is computed. It is possible to process the query promptly using this method. However, the ranking performance can be undermined by the deficiency of learning interaction between the query and the passage. In the case of the re-ranking task, the cross-encoder model was mainly used for the performance [8], [16], [19]. A model in this paper also has an architecture based on the cross-encoder method. The dual-encoder method is utilized for a full-ranking task processing a large number of passages in the corpus. Recently, there have been several works about efficient ranking using the dual-encoder method [20], [21]. However, we do not cover that architecture in detail in this paper.

C. BM25 AND PSEUDO RELEVANCE FEEDBACK

This paper proposes a term importance weighting method to fine-tune the ranking model. We adopt statistical methods traditionally used in information retrieval systems for deciding

whether each term is essential or not. BM25 is a probability model computing relevance between a given query and a passage. It is based on Poisson distribution that assumes two events cannot occur at the same time. The BM25 takes the appearance of a specific query term in a passage as an event. In the method, query terms are evaluated by the TF-IDF method. The probability that a specific query term appears in a passage becomes high as the passage length becomes longer. Therefore, the TF-IDF scores for query terms are normalized by the passage length in the BM25 method. The Eq. 1. is a formula of BM25 score. In the equation, a query Q with terms $\{t_1,\ t_2,\ \ldots, t_i\ \ldots\ t_n\}$ and a passage P are given. $TF\ (t_i, P)$ is the term frequency score between a query term t_i and the passage. $IDF\ (t_i)$ is the inverse document frequency score of the term t_i. $L(P)$ is a value dividing the length of passage P in terms by the average passage length. Lastly, k and b are hyperparameters of the BM25 score model. [22] is recommended for a more detailed explanation about the BM25 scheme.

$$BM25\ (Q, P) = \sum_{i=1}^{n} IDF\ (t_i)\ \frac{TF\ (t_i, P) \cdot (k+1)}{TF\ (t_i, P) + k\{1 - b + b \cdot L(P)\}} \quad (1)$$

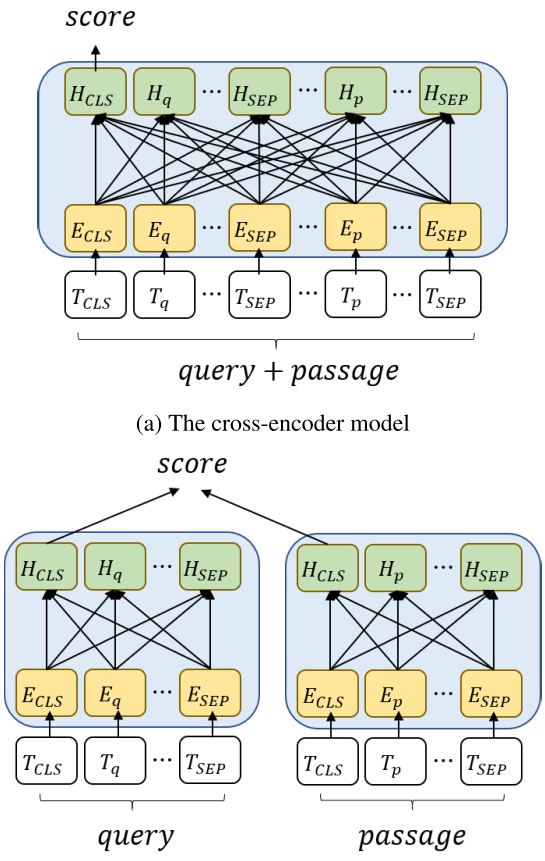
Pseudo relevance feedback is a method for query expansion utilizing search results from an initial query. In this term distribution-based method, passage terms satisfying the below condition are evaluated high for query expansion: the terms have to appear frequently in passages relevant to the initial query while hardly in non-relevant passages. There can be diverse approaches to the criteria for selecting relevant passages in information retrieval. In the case of pseudo relevance feedback, passages ranked high in the initial search results are judged as relevant. The Eq. 2. is used for the estimation of the passage term weight [23]. w_t is the relevance weight of a term t.

$$w_t = log\frac{p(1-q)}{q(1-p)} = log\frac{(r+0.5)(S-s+0.5)}{(R-r+0.5)(s+0.5)} \quad (2)$$

where p denotes the probability that the term t is assigned within the set of relevant sentences for the initial query, q the probability that the term is assigned withing the set of non-relevant sentences, r the number of relevant sentences with term t, R the number of entire relevant sentences, s the number of non-relevant sentences with term t, S the number of entire non-relevant sentences.

III. PROPOSED METHOD

In our work, a neural ranker based on the pre-trained language model learns the passage ranking and MLM tasks simultaneously as shown in Figure 1. In the MLM task, the proposed model is exposed to sequences with masked tokens and understands target corpus by predicting the original term tokens. The pre-trained language models tokenize each term in the input sequence to separated tokens, and we refer to the tokens as "term tokens" in this study. When the ranking



(a) The cross-encoder model

(b) The dual-encoder model

FIGURE 2. Comparison of the cross-encoder model and the dual-encoder model.

model is trained with the MLM task during the fine-tuning stage, the model estimates the relevance to a query using masked passages, which are created by selecting different masked tokens in each epoch; this allows the model to utilize the advantage of learning with more diverse text sequences without additional data. Thus, such a multitask-learning method emulates the effects of data augmentation. Figure 3. is an example of data augmentation using the proposed masking method. In the original ranking task, the model has to predict the relevance between a query "school location?" and a passage "next to the police office." In our method, the model is trained to predict relevance even if some parts of the passage are masked. The positions of tokens to be masked are different at each epoch while the model is exposed to diverse sequences.

A. BM25-BASED TERM IMPORTANCE ESTIMATION

In the MLM task, masking each term tokens with equal probability is a common approach for existing pre-trained language models. However, we suggest a new method masking the term tokens according to their importance weights. This method masks the less-important terms in the passage more frequently and avoids masking of each term token uniformly. It is assumed that masking contextually important terms

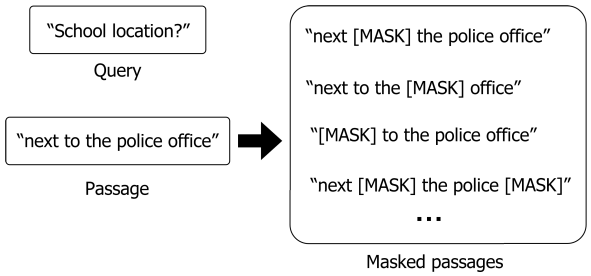


FIGURE 3. An example of data augmentation through the MLM task.

too often could undermines training because these terms commonly play key roles in retrieving relevant passages for the given queries. This new weighted-masking method is devised to improve the model's ranking performance. Given the terms $\{t_1, t_2, \ldots, t_i \ldots t_n\}$ in a passage P, we first calculate the BM25 score where $score(t_i)$ for the i-th term t_i, to compute the importance score.

$$score(t_i) = \frac{BM25(t_i) - \min_{t \in P} BM25(t)}{\max_{t \in P} BM25(t) - \min_{t \in P} BM25(t)} \tag{3}$$

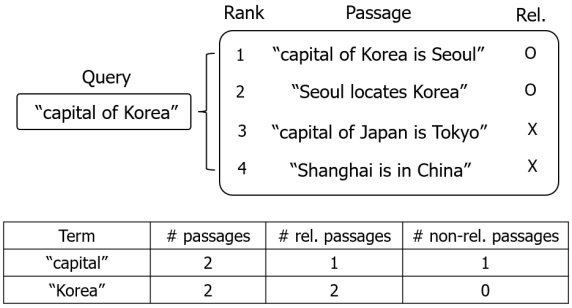
where $t \in P$ is one of every term in the given passage.

About 15% of the term tokens in each input passage are sampled for masking. The i-th term token t_i is sampled with the probability $P_{mask}(t_i)$ shown in Eq. 4. The probability each term token to be sampled increases as its importance score decreases, according to the following equation.

$$P_{mask}(t_i) = \frac{1 - score(t_i)}{\sum_{t \in P}(1 - score(t))} \tag{4}$$

Besides, we utilize embedding vector representing whether each passage sentence takes a core role in the passage or not. To calculate the importance weight of a sentence, we use the BM25 scores that are already calculated for the terms in the sentence and the sequential order of the sentence in the passage. The approach utilizing location of sentences is inspired by existing related studies. Ko *et al.* proposed a method weighting sentences at the beginning of a passage more for snippet generation in information retrieval [23]. Jeong *et al.* scored the passages at the beginning higher for selecting salient sentences in summarization system [24]. For a passage P with n sentences $\{s_1, s_2, \ldots, s_n\}$, the importance score of the k-th sentence with m terms $\{t_1, t_2, \ldots, t_i \ldots t_n\}$ is computed as shown in Eq. 5 and Eq. 6. The importance score is calculated by adding the sentence location weight to the sum of the normalized BM25 scores of the words in the sentence via min-max normalization. The sentence location weight is higher when the sentence is at the beginning of the passage. The sentences are then sorted by the scores, and about 10% of the sentences with the highest scores are selected as the core sentences. The ratio of core sentences is a hyperparameter, and we chose the ratio of 10% because of the best experiment result.

$$g(s_k) = \sum_{i=1}^{m} BM25(t_i) \tag{5}$$



Term	# passages	# rel. passages	# non-rel. passages
"capital"	2	1	1
"Korea"	2	2	0

FIGURE 4. An example of term importance estimation based on results from initial search.

$$score(s_k) = \frac{g(s_k) - \min_{s \in P} g(s)}{\max_{s \in P} g(s) - \min_{s \in P} g(s)} + \left(1 - \frac{k}{n}\right) \tag{6}$$

where t_i denotes the ith term in a passage, s_k the kth sentence in the passage, and $s \in P$ one of all sentences in a passage.

B. PSEUDO RELEVANCE FEEDBACK-BASED TERM IMPORTANCE ESTIMATION

Our BM25-based method mainly depends on term matching information. We suggest an additional term weighting method leveraging results from the initial search based on the pseudo relevance feedback. In Figure 4, the terms ''capital'' and ''Korea'' have the same importance score when evaluated by inverse document frequency. However, the term ''Korea'' has a higher probability of appearing in two relevant passages than the term ''capital'' when the search results are considered. In the case of non-relevant passages, the probability for the term ''Korea'' is zero. Therefore, it is possible to conclude that ''Korea'' is more important for the ranking task than ''capital.'' In the passage re-ranking task, candidate passages are provided for each query. Our method considers the candidate passages as results from the initial search in the pseudo relevance feedback. We utilize high-ranked passages in the candidates as relevant cases and low-ranked passages as non-relevant cases. Besides, we refer to [23] to calculate term importance weights based on the pseudo relevance feedback.

Given a query Q and a passage P with terms $\{t_1, t_2, \ldots, t_i \ldots t_n\}$, the importance weight $score_{PRF}(Q, t_i)$ for the ith term t_i is as following equation. Among the candidate passages for Q, we consider passages ranked from 1st to kth as relevant and those under kth as non-relevant. For the input sequence, 15% of term tokens are sampled for masking. The probability that each token is sampled is proportional to the importance score $score_{PRF}(Q, t_i)$.

$$PRF(Q, t_i) = log \frac{(r + 0.5)(S - s + 0.5)}{(R - r + 0.5)(s + 0.5)} \tag{7}$$

where r denotes the number of relevant passages with term t_i, R the number of entire relevant passages, s the number of non-relevant passages with term t_i, S the number of entire

non-relevant passages.

$$score_{PRF}(Q, t_i) = \frac{softmax(BM25(t_i)) + softmax(PRF(Q, t_i))}{2} \quad (8)$$

C. SELF-SUPERVISED FINE-TUNING WITH MLM

Our architecture includes a pre-trained language model as a shared layer, a regression layer $W_{RP} \in \mathbb{R}^{1\times H}$ for the ranking task, and another layer $W_{MLM} \in \mathbb{R}^{V\times H}$ for the MLM task, as described in Figure 1. We especially utilized RoBERTa [11] as the pre-trained language model in the experiment. However, other models are also acceptable in our architecture. Pre-trained language models represent input sequence as hidden vector. Besides, the number of vocabularies in the language models needs to be defined before training. In the case of our notation, V is the number of vocabularies and H indicates the dimension of the hidden vector. For the input sequence to the model, a query and a passage are concatenated following the input format of RoBERTa with a [CLS] token at the head and [SEP] token at the ends. The hidden vector from the shared layer is entered into layers for ranking and MLM task separately. Each layer calculates training loss value and the two task losses are combined to total loss L_{total} as in Eq. 9. In the equation, a hyperparameter λ is used to manipulate the impact of the MLM task on training the ranking model.

$$L_{total} = L_{RP} + \lambda \cdot L_{MLM} \quad (9)$$

In the case of information retrieval systems, loss functions can be categorized into three types: pointwise, pairwise, and listwise approach. The proposed model is trained by the listwise approach that optimizes a ranking for several input passages to one given query. In this method, a query and several candidate passages are given and each query–passage pair is entered into the model as a concatenated sequence. When H_{CLS} indicates the hidden vector of [CLS] token from the sequence, the cross-entropy loss[1] is minimized according to Eq. 10. for each query–passage pair x. In this notation, X is the set of pairs with the relevance label $y \in 0, 1$. The value of cross-entropy loss becomes higher as a difference between the value of actual label and the expected one increases. Therefore, the model has to be back-propagated in a way to minimize the loss value.

$$f(x) = softmax\left(H_{CLS} W_{RP}^T\right), \quad L_{RP} = -\sum_{x\in X} y \log f(x) \quad (10)$$

IV. EXPERIMENTS

A. DATASETS

MS MARCO [15] is an information retrieval dataset sampled from large-scale query logs of the search engine Microsoft Bing. It includes about 8.8 million text passages extracted from web documents. The relevance labels between queries and passages were annotated manually by human editors. The results from the dataset can be evaluated through online leaderboard submissions. The passage ranking leaderboard includes full ranking task and re-ranking task. The leaderboard organizers officially provide 1,000 candidate passages for each query for the re-ranking task. For more detailed information about the dataset, please refer to [15].

[1]The cross-entropy loss is one of the most popular loss functions in machine learning studies. For more detailed explanations about the cross-entropy loss function, chapter 4.3 of [25] is available.

B. EXPERIMENTAL SETTINGS

In the case of pre-trained language models for the proposed architecture, both base and large versions of RoBERTa were used in the experiments. The learning rate of the Adam optimizer was 5e-5 for the base model and 1e-5 for the large model. One TITAN X GPU was used for the base model, and one Tesla V100 GPU was for the large model. The training batch size was 16, and the maximum input length was 512. One positive passage and seven negative passages to the given query were used for calculating the listwise loss. Fine-tuning was performed without additional pre-training steps, and the development set performance was evaluated at the end of each epoch. The final value of the hyperparameter λ, which is the proportion of MLM loss in the total loss, was 1. In the case of the term weighing method using pseudo relevance feedback, hyperparameter k divides relevant and non-relevant passages among the candidates for each query. 100 is selected for the value of hyperparameter k. The BM25 scores for the experiments were calculated by Pyserini, a Python interface of the Lucene-based retriever Anserini [26].

The evaluation metrics were MRR@10 and Mean Rank (MR). MR was used only for the development set because the test results are only provided as the total MRR@10 scores. In the case of methods with pseudo relevance feedback, Hits@k was additionally used for evaluation. It indicates whether relevant passages ranked among the top-k retrieved results. We adopted a listwise ranking model without MLM fine-tuning as the baseline. The MS MARCO is a massive dataset that requires about a week for training and test with our experimental settings. Therefore, we sampled 10% of the training set and 2,000 queries in the development set (about 30%) for baseline comparison with the proposed model. It was enough amount for proving our method achieved significant improvement over the baseline. However, the final model for leaderboard submission was trained and evaluated with entire dataset for the performance at maximum.

V. RESULTS AND ANALYSIS

A. COMPARISON OF BASELINE AND BM25-BASED MLM MODELS

Table 1 was comparison results between the baseline and proposed models using BM25 for the MLM task. $LR_{Baseline}$ was the listwise ranking baseline trained by fine-tuning without MLM. $LR + MLM$ was the model fine-tuned with both ranking and random MLM tasks. The term tokens in the input passage were sampled for masking with a probability

TABLE 1. Comparisons between the baseline and BM25-based MLM models using 10% of the training set. "*" indicates a *t*-test result that a *p*-value between the proposed model and the baseline is lower than 0.01.

Model (with $RoBERTa_{base}$)	Dev. Set	
	MRR@10	MR
$LR_{Baseline}$	0.329	22.228
$LR + MLM$	0.356 (+0.027)	15.716 (-6.512)
$LR + WMLM$	0.363 (+0.034)	15.312 (-6.916)
$LR + WMLM + SE$*	**0.364 (+0.035)**	**14.932 (-7.296)**

TABLE 2. Comparisons among models with the different proportion of the MLM task using 10% of the training dataset.

Model (with $RoBERTa_{base}$)	Dev. set	
	MRR@10	MR
$\lambda = 2.0$	0.362 (-0.002)	15.015 (+0.083)
$\lambda = 1.5$	0.361 (-0.003)	15.317 (+0.385)
$\lambda = 1.0$	**0.364**	**14.932**
$\lambda = 0.7$	0.364 (-0.000)	14.943 (+0.011)
$\lambda = 0.5$	0.364 (-0.000)	14.949 (+0.017)
$\lambda = 0.3$	0.360 (-0.004)	15.469 (+0.537)
$\lambda = 0.1$	0.357 (-0.007)	15.196 (+0.264)

of 15%. *LR + WMLM* used the weighted MLM task for fine-tuning instead of the random MLM. The importance score of each term token was computed based on the BM25 score. The *LR + WMLM + SE* model used input sequences with sentence importance embeddings. Table 1 showed that the MRR@10 score was improved by 3.5% points from the baseline when the weighted MLM was adopted for fine-tuning. Additionally, the approach with sentence embedding was helpful for improving MR. We used the paired *t*-test with a *p*-value threshold of 0.01, where * indicates a significant improvement over the baseline model.

B. THE PROPORTION OF THE MLM TASK

The hyperparameter λ in Eq. 9 was used to determine the proportion of the MLM task for the entire training loss. When the hyperparameter value was high, it was available to increase the impact of data augmentation. On the other side, decreasing the value of the hyperparameter enlarged the proportion of the ranking task and let the model focus on the original task. Table 2 demonstrated how the performance for the development set changed according to the value of the hyperparameter. Our model achieved at the best performance of both MRR@10 and MR when the hyperparameter value was 1. The decrease of the hyperparameter value also made the decrease of performance. It presented that the impact of data augmentation was significant even though the original task was less trained on to the model during the multitask learning.

C. COMPARISON WITH POST-TRAINING

Post-training is one of the self-supervised learning methods using the corpus about a specific domain or task [9]. It mainly uses the MLM task and the next sentence prediction (NSP) task just like the pre-training method. However, the post-training method is for learning frequent language patterns in a certain domain or task, while the pre-training method enhances the broad range of language understanding. Our work is similar to the previous works with the post-training method in that self-supervised learning such as MLM is used to train a certain task. However, our proposed model does not need any additional training steps before the fine-tuning stage, and it is different from the post-training models.

TABLE 3. Comparisons between the proposed model and the post-trained models using 10% of the training set for fine-tuning.

Model (with $RoBERTa_{base}$)	Dev. set	
	MRR@10	MR
$LR_{Baseline}$	0.329	22.228
$LR + WMLM + SE$	**0.364 (+0.035)**	**14.932 (-7.296)**
$LR + PT_{1epoch}$	0.335 (+0.006)	19.731 (-2.497)
$LR + PT_{2epoch}$	0.337 (+0.008)	18.728 (-3.500)
$LR + PT_{3epoch}$	0.327 (-0.002)	19.591 (-2.637)

To compare the post-training method and the proposed fine-tuning method in this paper, we post-trained RoBERTa base model using MS MARCO passages and fine-tuned the listwise ranking task on the model. Table 3 demonstrated the comparison of the performances between the post-trained model (*LR + PT*) and our model without post-training (*LR + WMLM + SE*). We used only the random MLM task for the post-training according to the training method of RoBERTa. In Table 3, executing post-training for one epoch meant exposing 8.8 million MS MARCO passages to the model once. The experiment results showed that the ranking model fine-tuned after post-training achieved better performance than the baseline model. However, this improvement was insignificant when compared with the result from our proposed model. The proposed model obtained MRR@10 performance 3.5% points higher than the baseline, while the performance from the post-trained model was 0.8% point higher than that of the baseline at the maximum.

D. COMPARISON OF BM25 AND PSEUDO RELEVANCE FEEDBACK FOR MASKING

We proposed an additional term weighting method using pseudo relevance feedback. The method utilized information from both relevant and non-relevant passages to a given query. We calculated the importance score according to Eq. 8 and more frequently masked terms with high score values. In Table 4, $LR + WMLM_{PRF}$ meant a model with MLM strategy using pseudo relevance feedback. $LR + WMLM_{PRF} + SE$ denoted a model utilizing sentence embedding and the MLM method with pseudo relevance feedback simultaneously. Those models obtained significant performances at Mean Rank and Hits compared to models using only BM25. It meant that information from pseudo relevance feedback made relevant passages ranked higher among many candidates. The key difference between the two metrics and MRR@10 was that they evaluate the quality of the entire ranking while passages under the 10th

TABLE 4. Comparisons between the models based on BM25 and pseudo relevance feedback using 10% of the training set for fine-tuning.

Model (with $RoBERTa_{base}$)	Dev set.					
	MRR@10	MR	Hits@5	Hits@10	Hits@20	Hits@50
$LR_{Baseline}$	0.329	22.228	0.478	0.579	0.665	0.744
$LR + WMLM$	0.363	15.312	0.529	0.625	0.705	0.774
$LR + WMLM + SE$	**0.364**	14.932	0.532	0.624	0.706	0.772
$LR + WMLM_{PRF}$	0.363	13.973	**0.534**	0.628	0.704	0.778
$LR + WMLM_{PRF} + SE$	0.363	**13.815**	0.530	**0.637**	**0.707**	**0.778**

TABLE 5. Comparisons among models with different masking priorities using 10% of the training dataset.

Model (with $RoBERTa_{base}$)	Dev. set	
	MRR@10	MR
$LR_{Baseline}$	0.329	22.228
$LR + WMLM_{DOWN}$	**0.363 (+0.034)**	**15.312 (-6.916)**
$LR + WMLM_{MEAN}$	0.361 (+0.032)	15.371 (-6.857)
$LR + WMLM_{UP}$	0.356 (+0.027)	15.387 (-6.841)

TABLE 6. Processing time for a query-passage pair.

Model Base: $RoBERTa_{base}$ Large: $RoBERTa_{large}$	Dev. set
	sec/query-passage
$LR_{Baseline}$ (Base)	7.43×10^{-3}
$LR + MLM$ (Base)	$\mathbf{7.46}\times10^{-3}$
$LR + WMLM + SE$ (Base)	7.52×10^{-3}
$LR + WMLM + SE$ (Large)	11.31×10^{-3}

rank were not considered with MRR@10. Even though the BM25-based model achieved slightly higher performance at MRR@10, new models with pseudo relevance feedback performed significantly better when considering the entire ranking or more ranking was important, as they did at MR and Hits.

E. COMPARISON WITH MASKING PRIORITIES

The comparison between three weighted MLM models with different masking strategies demonstrated that masking less significant terms was helpful for training. In Table 5, $LR + WMLM_{DOWN}$ was a model with the important terms being masked less frequently, while $LR + WMLM_{UP}$ had the important terms masked more frequently. $LR + WMLM_{MEAN}$ computed the average value of the token weights in each passage and masked terms near the average frequently. In all three models, the proportion of the total masked tokens were 15%. The MRR@10 score from $LR + WMLM_{DOWN}$ was the highest while one from $LR + WMLM_{UP}$ was the lowest.

F. EFFICIENCY OF TEST TIME

The proposed model did not require the MLM task during evaluation as same as in the baseline. Table 6 compared the query latencies by in terms of test time (seconds per a query-passage) during test. Even though adding sentence embedding made test time a little longer, there was nearly no difference in the latency between the baseline and our models.

TABLE 7. Comparisons between the proposed and other models in MS MARCO leaderboard using the total training set.

Model Base: $RoBERTa_{base}$ Large: $RoBERTa_{large}$	Dev. set	Test set
	MRR@10	MRR@10
DUET V2 [27]	0.252	0.253
OpenMatch - ELECTRA Base [28]	0.352	0.344
OpenMatch - ELECTRA Large [28]	0.388	0.376
TF-Ranking Ensemble [19]	0.405	0.391
$LR_{Baseline}$ (Base)	0.345	-
$LR + MLM$ (Base)	0.382	-
$LR + WMLM + SE$ (Base)	0.383	**0.372**
$LR + WMLM + SE$ (Large)	0.389	**0.378**

G. COMPARISON WITH THE LEADERBOARD MODELS

For the MS MARCO leaderboard submission, we selected a model with BM25-based MLM and sentence embedding. Models using pseudo relevance feedback demonstrated better performance at other metrics. However, we submitted a model with the best performance at MRR@10, which was the official metric of the leaderboard. In the MS MARCO passage re-ranking leaderboard, our model outperformed the state-of-the-art, with the exception for an ensemble-based model of BERT, ELECTRA, and RoBERTa [19] from Google Research. Even though this model used the combination of three large-scale neural language models, difference between the MRR@10 scores of the ensemble model and the proposed model as a single model was only 1.3% point. The performance records from the leaderboard are summarized in Table 7.[2] DUET V2 [27] was the official baseline of the leaderboard. OpenMatch [28] was a library for several tasks in IR, whose implementation of ELECTRA Large was the best re-ranking model except for the ensemble-based model [19] before the proposed model was submitted. It was interesting that our model with the base version of RoBERTa also achieved significant performance as described in Table 7. High performance and fast query latency were available simultaneously given that the base version of RoBERTa required lower computational cost than the models of larger version.

VI. CONCLUSION AND FUTURE WORK

This study proposes a fine-tuning method using the MLM task for neural re-ranking models. The importance scores for

[2]The leaderboard records in this paper are based on the ranking of June 8, 2021. Those results are posted on the website https://microsoft.github.io/MSMARCO-Passage-Ranking-Submissions/leaderboard/. Our model's name in the leaderboard is "SSFT", an abbreviation of self-supervised fine-tuning. The team name of the model is anonymous.

the term tokens and sentences are estimated for selecting the tokens to be masked and core sentences in the model. The goal of the proposed model is to sustain balance between the ranking performance and efficiency. The ranking performance of the proposed model is the state-of-the-art in the MS MARCO passage re-ranking leaderboard among single models. We analyzed the data augmentation effect and time efficiency of the proposed model by various experiments. This study expands our previous work by utilizing pseudo relevance feedback method for calculating term importance, which is more adaptive approach to the ranking system.

Self-supervised learning in the fine-tuning phase can be applied for other tasks in natural language processing and IR using pre-trained language models. More discussions to develop efficient neural models using self-supervised learning would be available in the future.

REFERENCES

[1] R. Herbrich, T. Graepel, and K. Obermayer, "Support vector learning for ordinal regression," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, Edinburgh, U.K., 1999, pp. 97–102.
[2] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 89–96.
[3] C. Burges, R. Ragno, and Q. Le, "Learning to rank with nonsmooth cost functions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, 2006, pp. 193–200.
[4] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao, "Adapting boosting for information retrieval measures," *Inf. Retr.*, vol. 13, no. 3, pp. 254–270, 2010.
[5] B. Mitra, F. Diaz, and N. Craswell, "Learning to match using local and distributed representations of text for web search," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 1291–1299.
[6] C. Xiong, Z. Dai, J. Callan, Z. Liu, and R. Power, "End-to-end neural ad-hoc ranking with kernel pooling," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 55–64.
[7] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang, "RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Human Lang. Technol.*, 2021, pp. 5835–5847.
[8] R. Nogueira and K. Cho, "Passage re-ranking with BERT," 2019, *arXiv:1901.04085*.
[9] X. Sun, H. Tang, F. Zhang, Y. Cui, B. Jin, and Z. Wang, "TABLE: A task-adaptive BERT-based listwise ranking model for document retrieval," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 2233–2236.
[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2019, pp. 4171–4186.
[11] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.
[12] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, "Don't stop pretraining: Adapt language models to domains and tasks," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 8342–8360.
[13] S. E. Robertson and S. Walker, "Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval," in *Proc. SIGIR*. London, U.K.: Springer, 1994, pp. 232–241.
[14] C. Buckley, G. Salton, and J. Allan, "The effect of adding relevance information in a relevance feedback environment," in *Proc. SIGIR*. London, U.K.: Springer, 1994, pp. 292–300.
[15] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and Li Deng, "MS MARCO: A human generated machine reading comprehension dataset," in *Proc. CoCo@ NIPS*, 2016, pp. 1–10.
[16] M. Kim and Y. Ko, "Self-supervised fine-tuning for efficient passage re-ranking," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 3142–3146.
[17] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–19.
[18] L. Boualili, J. G. Moreno, and M. Boughanem, "MarkedBERT: Integrating traditional IR cues in pre-trained language models for passage retrieval," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1977–1980.
[19] S. Han, X. Wang, M. Bendersky, and M. Najork, "Learning-to-rank with BERT in TF-ranking," 2020, *arXiv:2004.08476*.
[20] O. Khattab and M. Zaharia, "ColBERT: Efficient and effective passage search via contextualized late interaction over bert," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2020, pp. 39–48.
[21] L. Gao, Z. Dai, and J. Callan, "COIL: Revisit exact lexical match in information retrieval with contextualized inverted list," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2021, pp. 3030–3042.
[22] S. Robertson and H. Zaragoza, *The Probabilistic Relevance Framework: BM25 and Beyond*. Delft, The Netherlands: Now Publishers, 2009.
[23] Y. Ko, H. An, and J. Seo, "An effective snippet generation method using the pseudo relevance feedback technique," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2007, pp. 711–712.
[24] H. Jeong, Y. Ko, and J. Seo, "How to improve text summarization and classification by mutual cooperation on an integrated framework," *Expert Syst. Appl.*, vol. 60, pp. 222–233, Oct. 2016.
[25] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2007.
[26] P. Yang, H. Fang, and J. Lin, "Anserini: Enabling the use of Lucene for information retrieval research," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 1253–1256.
[27] B. Mitra and N. Craswell, "An updated duet model for passage re-ranking," 2019, *arXiv:1903.07666*.
[28] Z. Liu, K. Zhang, C. Xiong, Z. Liu, and M. Sun, "OpenMatch: An open source library for Neu-IR research," 2021, *arXiv:2102.00166*.

MEOUNGJUN KIM is currently pursuing the M.S. degree with the Department of Artificial Intelligence, Sungkyunkwan University, Suwon, South Korea. His research interests include natural language processing and information retrieval.

YOUNGJOONG KO received the Ph.D. degree from the Department of Computer Science, Sogang University, Seoul, South Korea, in 2003. He worked at LG-EDS, from 1996 to 1997. Since 2004, he joined the Faculty of Dong-A University, Busan, where he leaded the Intelligent System Laboratory, Department of Computer Engineering. He worked as a Visiting Scholar at the Computational Linguistics and Information Processing Laboratory (CLIP), University of Maryland, College Park, MD, USA, from 2011 to 2012. In 2019, he moved Sungkyunkwan University, Suwon-si. His research interests include natural language processing, machine learning (deep neural networks), spoken dialogue systems, question answering systems, knowledge graph engineering, information retrieval, and big data analysis.