# SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval


Yang Bai*† Tsinghua University

Xiaoguang Li* Huawei Noah's Ark Lab

Gang Wang Huawei Noah's Ark Lab

Chaoliang Zhang Huawei Noah's Ark Lab

Lifeng Shang Huawei Noah's Ark Lab

Jun Xu Renmin University of China

Zhaowei Wang Huawei Noah's Ark Lab

Fangshan Wang Huawei Technologies Co., Ltd

Qun Liu Huawei Noah's Ark Lab





## ABSTRACT

Term-based sparse representations dominate the first-stage text retrieval in industrial applications, due to its advantage in efficiency, interpretability, and exact term matching. In this paper, we study the problem of transferring the deep knowledge of the pre-trained language model (PLM) to **Term**-based **Spa**rse **r**epresentations, aiming to improve the representation capacity of bag-of-words(BoW) method for semantic-level matching, while still keeping its advantages. Specifically, we propose a novel framework SparTerm to directly learn sparse text representations in the full vocabulary space. The proposed SparTerm comprises an importance predictor to predict the importance for each term in the vocabulary, and a gating controller to control the term activation. These two modules cooperatively ensure the sparsity and flexibility of the final text representation, which unifies the term-weighting and expansion in the same framework. Evaluated on MSMARCO dataset, SparTerm significantly outperforms traditional sparse methods and achieves state of the art ranking performance among all the PLM-based sparse models.



## KEYWORDS

Fast Retrieval, Sparse Representation, BERT


## 1 INTRODUCTION

Text retrieval in response to a natural language query is a core task for information retrieval (IR) systems. Most recent work has adopted a two-stage pipeline to tackle this problem, where an initial set of documents are firstly retrieved from the document collection by a fast retriever, and then further re-ranked by more sophisticated models.

For the first-stage retrieval, neural dense representations show great potentials for semantic matching and outperform sparse methods in many NLP tasks, but this is not necessarily true in scenarios that emphasize long document retrieval and exact matching[9]. Moreover, for extremely large (e.g. 10 billion) candidates collection, the dense method has to struggle with the efficiency vs. accuracy tradeoff. Classical term-based sparse representations, also known as bag-of-words (BoW), such as TF-IDF [15] and BM25 [14], can efficiently perform literal matching, thus playing a core role in industrial IR systems. However, traditional term-based methods are generally considered to have insufficient representation capacity and inadequate for semantic-level matching.

| Query | Can hives be a sign of pregnancy? | |
|---|---|---|
| Type | Term frequency | SparTerm |
| Literal term Weights | hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine | hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine |
| Term expansion | | symptoms:1.0, women:0.99, rash:0.98, feel:0.99, causing:0.97, body:0.96, affect:0.96, baby:0.94, pregnant:0.93, **sign:0.91**, … |

**Figure 1: The comparison between BoW and SparTerm representation. The depth of the color represents the term weights, deeper is higher. Compared with BoW, SparTerm is able to figure out the semantically important terms and expand some terms not appearing in the passage but very semantically relevant, even the terms in the target query such as "sign".**

Some attempts have been made to make sparse methods beyond lexical matching while still keeping their advantages. SRNM [17] learns latent sparse representations for the query and document based on dense neural models, in which the "latent" token plays the role of the traditional term during inverted indexing. One challenge about SNRM is that it loses the interpretability of the original terms, which is critical to industrial systems.

Recently proposed pre-trained language models(PLM) such as ELMO [12] and BERT [4] show superior performance in many NLP tasks, thus providing new opportunities to transfer deep contextualized knowledge from dense representations to sparse models. Focusing on the relevant relationship between a passage/document and

*Both authors contributed equally to this research.

†This work is done when Yang Bai is an intern at Huawei Noah's Ark Lab.

corresponding query, DeepCT [2] and Doc2Query [11] learn PLM-based models to enhance the performance of traditional BoW methods. The difference is that DeepCT learns a regression model to re-weight terms with contextualized representations, while Doc2query learns an encoder-decoder generative model to expand query terms for passage. Both of these two methods train an auxiliary intermediate model and then help refine the final sparse representations to achieve better text ranking performance.

In this paper, we propose a novel framework SparTerm to learn **Term**-based **Spa**rse **r**epresentations directly in the full vocabulary space. Equipped with the pre-trained language model, the proposed SparTerm learns a function to map the frequency-based BoW representation to a sparse term importance distribution in the whole vocabulary, which offers the flexibility to involve both term-weighting and expansion in the same framework. As shown in Figure 1, compared with BoW representation, SparTerm assigns more weights to the term of high distinguishability given the context, and expand extra terms hopefully bridging the lexical gap with future queries. We empirically show that SparTerm significantly increase the upper limit of sparse retrieval methods, and gives new insights of transferring deep knowledge from PLM-based representation to simple BoW representations.

More specifically, SparTerm comprises an importance predictor and a gating controller. The importance predictor maps the raw input text to a dense importance distribution in the vocabulary space, which is different from traditional term weighting methods that only consider literal terms of the input text. To ensure the sparsity and flexibility of the final representation, the gating controller is introduced to generate a binary and sparse gating signal across the dimension of vocabulary size, indicating which tokens should be activated. These two modules cooperatively yield a term-based sparse representation based on the semantic relationship of the input text with each term in the vocabulary.

**Our contributions.** In summary, we propose to directly learn term-based sparse representation in the full vocabulary space. The proposed SparTerm indicates that there is much space for improving the ranking performance of termed-based representations, while still keeping the interpretability and efficiency of BoW methods. Evaluated on MSMARCO [10] dataset, SparTerm significantly outperforms previous sparse models based on the comparable size of PLMs. The top-ranking performance of SparTerm even outperforms Doc2Query-T5, which is based on the pre-trained model of 2x model size and 70x pre-training corpus size. Moreover, we conduct further empirical analysis about how the deep knowledge of PLMs can be transferred to the sparse method, which gives new insights for sparse representation learning.

## 2 RELATED WORK

Our work relates to two research fields: bag-of-words representations and pre-trained language model for text retrieval.

### 2.1 Bag-of-words Methods

Bag-of-words(BoW) methods have played a central role in the first-stage retrieval. These methods convert a document or query into a set of single terms, and each term associates a weight to characterize its weight. Most of the early common practice adopted TF-IDF style models to calculate weights. Robertson [14] proposed the well-known method BM25, which further improve the performance of the original TF-IDF. Later proposed methods, such as [7], [18], [16], did not show much advantage over BM25. More recently, Hamed Zamani [17] proposed SRNM to learn a sparse coding in hidden space using weak supervision, which shows good potential for solving the "lexical mismatch" problem. However, the latent unexplainable tokens can not ensure that documents with exact matched terms can be retrieved.

### 2.2 PLMs for dense text retrieval

The pre-trained language models like BERT [4] show new possibilities for text retrieval. Based on dense representations, Lee [8] proposed ORQA with bi-encoder architecture to retrieve candidate passages for question answering using FAISS [5]. However, analysis from [9] concludes that bi-encoders based on dense representation suffer from its capacity limitation in scenarios that emphasize long document retrieval and exact matching. Following the late-interaction paradigm, Khattab [6] proposed Col-BERT to conduct efficient interaction between the query and document, which can run 150x faster than fully-interactive BERT but achieve comparable precision. Though much faster than BERT, Col-BERT is still not computationally feasible for large scale first-stage retrieval, for the existence of the late interaction layer.

### 2.3 PLMs for sparse text retrieval

Several PLM-based models have emerged to improve the traditional sparse BoW representations. Dai [2] proposed DeepCT to estimate a term's weight considering its contextualized information, and this work was later extended to generate document-level term weights [3]. Another work Doc2query [11] tries to "translate" potential queries to expand document content, which also shows a large improvement compared to the traditional BM25 method. The biggest difference between our work and these two methods is that DeepCT and Doc2Query train an auxiliary intermediate model to help refine the sparse representations, while SparTerm is desinged to directly learn sparse representations within the whole vocabulary.

## 3 SPARSE REPRESENTATION LEARNING

This section presents the model architecture of SparTerm and the corresponding training strategy.

### 3.1 Overview

Figure 2(a) depicts the general architecture of SparTerm which comprises an importance predictor and a gating controller. Given the original textual passage $p$, we aim to map it into a deep and contextualized sparse representation $p'$ in the vocabulary space. The mapping process can be formulated as:

$$p' = \mathcal{F}(p) \odot \mathcal{G}(p) \tag{1}$$

where $\mathcal{F}$ is the item importance predictor and $\mathcal{G}$ the gating controller. The importance predictor $\mathcal{F}$ generates a dense vector representing the semantic importance of each item in the vocabulary. The gating controller $\mathcal{G}$ generates a binary gating vector to control which terms to appear in the final sparse representation. To achieve

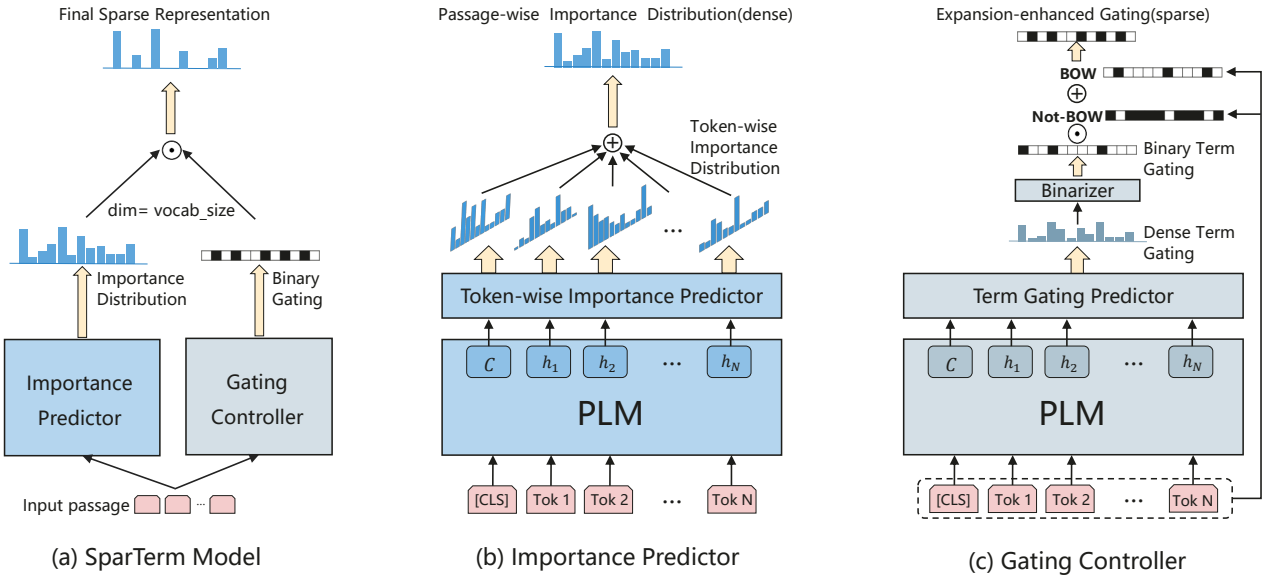


**Figure 2: Model Architecture of SparTerm. Our overall architecture contains an importance predictor and a gating controller. The importance predictor generates a dense importance distribution with the dimension of vocabulary size, while the gating controller outputs a sparse and binary gating vector to control term activation for the final representation. These two modules cooperatively ensure the sparsity and flexibility of the final representation.**

this, we let $||\mathcal{G}(p)|| < \lambda$ and $\mathcal{G}(p) \in \{0,1\}^v$, where $\lambda$ is the maximum number of non-zero elements for $p'$, and $v$ the vocabulary size. These two modules cooperatively ensure the sparsity and flexibility of the final representation $p'$. We discuss the detailed model architecture and learning strategy for $\mathcal{F}$ and $\mathcal{G}$ in the following sections.

## 3.2 The Importance Predictor

Given the input passage $p$, the importance predictor outputs semantic importance of all the terms in the vocabulary, which unify term weighting and expansion into the framework. As shown in Figure 2(b), prior to importance prediction, BERT-based encoder is employed to help get the deep contextualized embedding $h_i$ for each term $w_i$ in the passage $p$. Each $h_i$ models the surrounding context from a certain position $i$, thus providing a different view of which terms are semantically related to the topic of the current passage. With a token-wise importance predictor, we obtain a dense importance distribution $I_i$ of dimension $v$ for each $h_i$:

$$I_i = Transform(h_i)E^{\mathrm{T}} + b \tag{2}$$

where $Transform$ denotes a linear transformation with GELU activation and layer normalization, $E$ is the shared word embedding matrix and $b$ the bias term. Note that the token-wise importance prediction module is similar to the masked language prediction layer in BERT, thus we can initialize this part of parameters directly from pre-trained BERT. The final passage-wise importance distribution can be fetched simply by the summation of all token-wise importance distributions:

$$I = \sum_{i=0}^{L} Relu(I_i) \tag{3}$$

where $L$ is the sequence length of passage $p$ and Relu activation function is leveraged to ensure the nonnegativity of importance logits.

## 3.3 The Gating Controller

The gating controller generates a binary gating signal of which terms to activate to represent the passage. First, the terms appearing in the original passage, which we referred to as literal terms, should be activated by the controller by default. Apart from the literal terms, some other terms related to the passage topic are also expected to be activated to tackle the "lexical mismatch" problem of BOW representation. Accordingly, we propose two kinds of gating controller: literal-only gating and expansion-enhanced gating, which can be applied in scenarios with different requirements for lexical matching.

**Literal-only Gating.** If simply setting $\mathcal{G}(p) = BOW(p)$, where $BoW(p)$ denotes the binary BoW vector for passage $p$, we get the literal-only gating controller. In this setting, only those terms existing in the original passage are considered activated for the passage representation. Without expansion for non-literal terms, the sparse representation learning is reduced to a pure term re-weighting scheme. Nevertheless, in the experiment part, we empirically show that this gating controller can achieve competitive retrieval performance by learning importance for literal terms.

**Exapnsion-enhanced Gating.** The expansion-enhanced gating controller activates terms that can hopefully bridge the "lexical mismatch" gap. Similar to the importance prediction process formulated by Equation (2) and Equation (3), we obtain a passage-wise dense term gating distribution $G$ of dimension $v$ with independent network parameters, as shown in Figure 2(c). Note that although the

| Term expansion | Description and examples |
|---|---|
| Passage2query | Expand words that tend to appear in corresponding queries, i.e. "how far", "what causes". |
| Synonym | Expand synonym for original core words, i.e. "cartoon"->"animation". |
| Co-occurred words | Expand frequently co-occurred words for original core words, i.e. "earthquakes"->"ruins". |
| Summarization words | Expand summarization words that tend to appear in passage summarization or taggings. |

**Table 1: Different kinds of term expansion.**

gating distribution $G$ and the importance distribution $I$ share the same dimension $v$, they are different in logit scales and mathematical implications. $I$ represents the semantic importance of each term in vocabulary, while $G$ quantifies the probability of each term to participate in the final sparse representation. To ensure the sparsity of $p^{'}$, we apply a binarizer to $G$:

$$G^{'} = Binarizer(G) \tag{4}$$

where the *Binarizer* denotes a binary activation function which outputs only 0 or 1. The gating vector for expansion terms $G_e$ is obtained by:

$$G_e = G^{'} \odot (\neg BoW(p)) \tag{5}$$

where the bitwise negation vector $\neg BoW(p)$ is applied to ensure orthogonal to the literal-only gating. Simply adding the expansion gating and the literal-only gating, we get the final expansion-enhanced gating vector $G_{le}$:

$$G_{le} = G_e + BoW(p) \tag{6}$$

Involving both literal and expansion terms, the final sparse representation can be a "free" distribution in the vocabulary space. Note that in the framework of SparTerm, expanded terms are not directly appended to the original passage, but are used to control the gating signal of whether allowing a term participating the final representation. This ensures the input text to the BERT encoder is always the natural language of the original passage.

## 3.4 Training

In this section, we introduce the training strategy of the importance predictor and expansion-enhanced gating controller.

**Training the importance predictor.** The importance predictor is trained end-to-end by optimizing the ranking objective. Let $R = \{(q_1, p_{1,+}, p_{1,-}), ..., (q_N, p_{N,+}, p_{N,-})\}$ denote a set of N training instances; each containing a query $q_i$, a posotive candidate passage $p_{i,+}$ and a negative one $p_{i,-}$, indicating that $p_{i,+}$ is more relevant to the query than $p_{i,-}$. The loss function is the negative log likelihood of the positive passage:

$$L_{rank}(q_i, p_{i,+}, p_{i,-}) = -\log \frac{e^{sim(q_i^{'}, p_{i,+}^{'})}}{e^{sim(q_i^{'}, p_{i,+}^{'})} + e^{sim(q_i^{'}, p_{i,-}^{'})}} \tag{7}$$

where $q_i^{'}$, $p_{i,+}^{'}$, $p_{i,-}^{'}$ is the sparse representation of $q_i$, $p_{i,+}$, $p_{i,-}$ obtained by Equation (1), *sim* denotes any similarity measurement such as dot-product. Different with the training objective of DeepCT **??**, we don't directly fit the statistical term importance distribution, but view the importance as intermediate variables that can be learned by distant supervisory signal for passage ranking. End-to-end learning can involve every terms in the optimization process, which can yield smoother importance distribution, but also of enough distinguishability.

**Training the exapnsion-enhanced gating controller.** We summarize four types of term expansion in Table 1, all of which can be optimized in our SparTerm framework. Intuitively, the pre-trained BERT already has the ability of expanding synonym words and co-occured words by the Masked Language Model(MLM) pre-training task. Therefore, in this paper, we focus on expanding passage2query-alike and summarization terms. Given a passage-query/summary parallel corpus C, where $p$ is a passage, $t$ the corresponding target text, and $T$ of dimension $v$ is the binary bag-of-words vector of $t$. We use the binary cross-entropy loss to maximize probability values of all the terms in vocabulary:

$$L_{exp} = -\lambda_1 \sum_{j \in \{m|T_m=0\}} log(1 - G_j) - \lambda_2 \sum_{k \in \{m|T_m=1\}} log G_k \tag{8}$$

where $G$ is the dense gating probability distribution for $p$, $\lambda_1$ and $\lambda_2$ two tunable hyper-parameters. $\lambda_1$ is the loss weight for terms expected not to be expanded, while $\lambda_2$ is for terms that appear in the target text. In the experiment, we set $\lambda_2$ much larger than $\lambda_1$ to encourage more terms to be expanded.

**End-to-end joint training.** Intuitively, the supervisory ranking signal can also be leveraged to guide the training of the gating controller, thus we can train the importance predictor and gating controller jointly:

$$L = L_{rank} + L_{exp} \tag{9}$$

# 4 EXPERIMENTAL SETUP

## 4.1 Datasets and Metrics

We evaluate our method on **MSMARCO** [10] which consists of two benchmark datasets:

**MSMARCO Passage Retrieval dataset** is based on the public MSMARCO dataset with a collection of 8.8M passages from Web pages gathered from Bing's results to 1M real-world queries. Each query is associated with one or very few passages marked as relevant while no passage explicitly indicated as irrelevant. We build a small dev set for validating the full ranking performance instead of re-ranking by sampling the most relevant 1M passages to 1000 queries from the original passage ranking dev set with BM25.

**MSMARCO Document Retrieval dataset** is based on the source documents which contain the passages in the passage retrieval task. The dataset contains 367,013 documents and 367,013 queries for training set and 5,193 queries for dev set.

The original Dev Set of MSMARCO dataset is a re-ranking task, which is inconsistent with the retrieval task. Therefore, to find the best checkpoint of our model more accurately we build a new Dev Set to evaluate the retrieval performance by sampling about 1M passages from the collections and 1,000 queries from the original Dev Set (including the top 1000 passages of each query retrieved by

| Model | MRR@10 | R@10 | R@20 | R@50 | R@100 | R@200 | R@500 | R@1000 |
|---|---|---|---|---|---|---|---|---|
| BM25 | 18.6 | - | 49 | 60 | 69 | 75 | 82 | 85.71 |
| Doc2query | 21.5 | - | - | - | - | - | - | 89.1 |
| Doc2query-T5 | 27.7 | - | - | 75.6 | 81.89 | 86.88 | 91.64 | 94.7 |
| DeepCT | 24.3 | 49 | 58 | 69 | 76 | 82 | 86 | 91 |
| SparTerm(literal-only) | 27.46 | 51.05 | 60.21 | 71.55 | 78.28 | 83.27 | 88.33 | 91.16 |
| SparTerm(expansion-only) | 19.8 | 40.93 | - | 63.42 | 70.96 | 77.62 | 84.81 | 89.08 |
| SparTerm(expansion-enhanced) | **27.94** | 51.95 | 61.58 | 72.48 | 78.95 | 84.05 | 89.5 | 92.45 |

**Table 2: Performances of different models on Dev Set of MSMARCO Passage Retrieval dataset.**

BM25). To evaluate the full ranking performance of our model, we use the sparse representation of each document to build the inverted index and use the sparse representation of queries to retrieval topK relevant documents and measure the performance with MRR@10 and Recall from top10 to top1000.

## 4.2 Implementation

The Importance Predictor and Gating Controller of our model have the same architecture and hyper-parameters of BERT (12-layer, 768-hidden, 12-heads, 110M parameters) and do not share weights. We initialize the Importance Predictor with Google's official pre-trained $\mathrm{BERT}_{\mathrm{base}}$ model while the parameters of Token-wise Importance Predictor are initialized with the Masked Language Prediction layer of BERT. When using expansion-enhanced gating, the Gating Controller is also initialized with $\mathrm{BERT}_{\mathrm{base}}$. We fine-tune our model on the training set of MSMARCO passage retrieval dataset on 4 NVIDIA-v100 GPUs with a batch size of 128. During the fine-tuning, we first fine-tune the Gating Controller with Equation (8) for 50k iterations where $\lambda_1 = 1e-3$ and $\lambda_2 = 1$. Then we fix the parameters of the Gating Controller and fine-tune our SparTerm jointly for 100k iterations. We use Adam optimizer with the learning rate $2 \times 10^{-5}$. To ensure the sparsity, the threshold in the Binarizer in Equation (4) is set to 0.7. We do not fine-tune our model on the training set of document retrieval dataset but just use the model trained on the passage retrieval dataset for the document ranking.

## 4.3 Baselines and Experimental Settings

We compare our model with the following strong baselines which are all methods based on sparse representation . The former two focus on re-weighting while the latter two focus on document expansion:

- **BM25**[14] is a bag-of-words retrieval models with frequency-based signals to estimate the weights of terms in a text.
- **DeepCT**[2] is a deep contextualized term weighting model which maps the BERT's representations to term weightings for retrieval.
- **Doc2query**[11] is a document expansion method with Transformer that can expand documents with terms related to the documents' content.
- **Doc2query-T5**[1] is a document expansion method which utilizes more powerful T5 [13] language model to generate queries for document expansion.

We also evaluate three different settings of SparTerm for evaluation:

- **SparTerm(literal-only)** uses Importance Predictor with the Literal-only Gating which can also be seen as a term weighting model.
- **SparTerm(expansion-only)** uses the Expansion-enhanced Gating for passage expansion without term weighting. We just add the expanded words (weight of each word is 1) to the passages.
- **SparTerm(expansion-enhanced)** implements both Importance Predictor and Expansion-enhanced Gating for sparse representation of passage.

# 5 EXPERIMENTAL RESULTS

## 5.1 Performance on Passage Full Ranking

Table 2 shows the full ranking performances of our models and baselines on MSMARCO Passage Retrieval dataset. SparTerm (expansion-enhanced) outperforms all baselines on MRR, achieving the state-of-the-art ranking performance among all sparse models, and outperforms all baselines except Doc2query-T5 on Recall. We find that SparTerm achieves more significant performance improvements on MRR and Recall@10-100, which illustrates that our model has a more significant ability on top ranking compared with previous sparse models. Further, pre-trained language model(PLM) based methods (DeepCT, Doc2query-T5, and SparTerm) perform better than those without PLM, demonstrating that PLM can facilitate the passage full ranking with better representation. Considering the improvements T5 brings to Doc2query, we believe that SparTerm can be further improved with more advanced PLM.

Even without any expansion, SparTerm(literal-only) outperforms DeepCT on both MRR and Recall, demonstrating that SparTerm can produce more effective term weights thus facilitating the retrieval. We also analyze the difference between SparTerm and DeepCT on term weighting in Section 5.4. With only the expanded words, SparTerm achieves a definite improvement compared with BM25, especially on Recall. This improvement proves the effectiveness of passage expansion on improving the Recall for retrieval.

## 5.2 Performance on Document Ranking

For the Document Ranking task, we cut down each document into several passages to adapt the max length (256) of the sequence of our model and generate the sparse representation of each passage with our model. We compare our models with two baseline methods: BM25 [14] and HDCT [3]. HDCT is based on the work of DeepCT and focuses on document ranking, which is also a term weighting method. HDCT compares two different ways to combine

| Model | MRR@10 |
|---|---|
| BM25+*PassageRetrievalMax* | 23.6 |
| HDCT+*PassageRetrievalMax* | 26.1 |
| BM25 | 24.5 |
| HDCT(sum) | 28.0 |
| HDCT(decay) | 28.7 |
| SparTerm(literal-only)+*PassageRetrievalMax* | 28.5 |
| SparTerm(expansion-enhanced)+*PassageRetrievalMax* | **29.0** |

**Table 3: Performance of baselines and our models on dev set of MSMARCO document ranking dataset. All use the max score of passages in the document as the document score at the query time.**

| Model | MRR@10 | R@1000 |
|---|---|---|
| Query-tf | 25.7 | 94.2 |
| Query-neural-symmetric | **26.4** | **94.7** |
| Query-neural-asymmetric | 25.4 | 94.2 |

**Table 4: Performances of our model with different query representation strategies on our new Dev Set of MSMARCO passage retrieval.**

the representations of passages for document ranking. The first one represents the document as a sum of the passage representations while the second one uses a decayed weighted sum. The *PassageRetrievalMax* does not represent the document but just calculates the scores of passages in the document and choose the maximum score as the score of the document for ranking. Table 3 shows the ranking performance of baselines and our models. Here we only report the results of *PassageRetrievalMax* of our models.

Strictly speaking, it is incomparable between HDCT and our models since we fine-tune SparTerm on MSMARCO pasage ranking dataset while HDCT was trained using document titles on MARCO. Even though, SparTerm(expansion-enhanced) still achieves a better performance on document ranking compared with HDCT, demonstrating that the sparse representation produced by SparTerm can also facilitate long document retrieval.

## 5.3 Comparison of Different Query Representation Methods

We conduct experiments to evaluate the performance of SparTerm with different query representation methods:

- **Query-tf** is a one-tower model that use tf-based vectors to represent the queries while use the model to represent documents.
- **Query-neural-symmetric** is a symmetric two-tower model to represent queries and passages that the two towers with the same architectures share the same weights.
- **Query-neural-asymmetric** is a asymmetric two-tower model that the two towers do not share weights. Queries and passages are represented with different towers.

The results are reported in Table 4, from where we find that the neural representation of queries with the symmetric two-tower model brings better performance on MRR and Recall on our built Dev set. The symmetric model performs better than the asymmetric model might because asymmetric two-tower architecture leads to twice the quantity of parameters, which makes the model more difficult to converge. We further analyze the distribution of passage term weights with different query representation methods and find that tf-based representation of query results in a sharper distribution compared to the neural representation. The reason may be that the query representation is fixed during training, the model needs to give more weights to the relative terms in the positive passage.

## 5.4 Analysis of Term Weighting

To further evaluate the ability of SparTerm on term weighting, we normalize the term weights of passages weighted by DeepCT and SparTerm(literal-only) to the same range and visulization them in Figure 3. Figure 3 shows three different queries(the first column) and the most relevant passages. The depth of the color represents the weights of terms, deeper is higher. We find that both DeepCT and SparTerm can figure out the most important terms and give them higher weights. However, DeepCT obtains sparser and sharper distributions and only activates very few terms in a passage, missing some important terms, such as "allergic reaction" in the first case. SparTerm can yield a smoother importance distribution by activating more terms though not appearing in the query. This distribution allows the passage to be retrieved by more queries. This also demonstrates that our model has a better ability on pointing out important terms in a passage.

## 5.5 Analysis of Term Expansion

Figure 3 shows the expanded terms and their probabilities for different passages predicted by the Gating Controller. The probability of each term illustrates how likely this term to be expanded. It is obvious that our model can really activate some important terms not appearing in the passage but very semantically similar, especially occurring in the queries such as "sign" in the first case and "temperature" in the second case.

In order to analyze how these words are expanded and which category in Figure 3 do they belong to, we trace the source of each expanded word and show the top 5 words with their logits which contribute to the expanded word in Figure 4. We can find that there are basically three different situations of the expanded terms:

**(1)** The passage2query terms such as "temperature": Almost every word in the passage contributes much to this kind of terms, which seem more likely to learn from the supervised signal.

**(2)** Synonyms of the original terms, i.e. "weather" and "climate", "rainfall" and "rain", "season, monthly" and "month", "heat" and "hot".

**(3)** Co-occurred words for the original terms, i.e. "season, heat"->"summer", "wet, humidity, weather"->"rain" and "heat, rainfall, humidity"->"tropical, monsoon".

The first situation is benefited by the optimization objective of the Gating Controller while the latter two are more likely the ability of MLM pretraining task since we reuse the MLM module for prediction in the Gating Controller.

| Query | Type | DeepCT | SPART |
|---|---|---|---|
| Can hives be a sign of pregnancy? | Literal term Weights | hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine . | hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine |
| | Term expansion | | symptoms:1.0, women:0.99, rash:0.98, feel:0.99, causing:0.97, body:0.96, affect:0.96, baby:0.94, pregnant:0.93, **sign:0.91**, … |
| Temperature in April in Bali | Literal term Weights | weather in bali in april with greatly reduced humidity levels ( around 65 % ) , april heralds the end of bali s wet season . monthly rainfall is reduced to 70 millilitres on average , the days are clearer and that classic bali heat is on the rise with some days reaching 33 c . | weather in bali in april with greatly reduced humidity levels ( around 65 % ) , april heralds the end of bali s wet season . monthly rainfall is reduced to 70 millilitres on average , the days are clearer and that classic bali heat is on the rise with some days reaching 33 c |
| | Term expansion | | month:0.99, rain:0.98, **temperature:0.97**, Java:0.97, summer:0.95, tropical:0.93, monsoon:0.93, hot:0.84, climate: 0.83… |
| Effects of detox juice cleanse | Literal term Weights | unhealthy weight loss . one of the positive side effects of a detox cleanse is weight loss . however , according to the same review article in obesity reviews , you ' re not losing fat weight on such a severe diet , but precious muscle mass | unhealthy weight loss . one of the positive side effects of a detox cleanse is weight loss . however , according to the same review article in obesity reviews , you ' re not losing fat weight on such a severe diet , but precious muscle mass |
| | Term expansion | | benefits:0.99, good:0.99, bad:0.99, harmful:0.98, vitamin:0.97, drugs:0.93, body:0.98, cause:0.97, definition:0.85, heavy:0.84 |

**Figure 3: Term weightings of different passages weighted by DeepCT and SparTerm, and the expanded terms with their probabilities (before the binarization) predicted by SparTerm. The depth of the color represents the term weights, deeper is higher.**

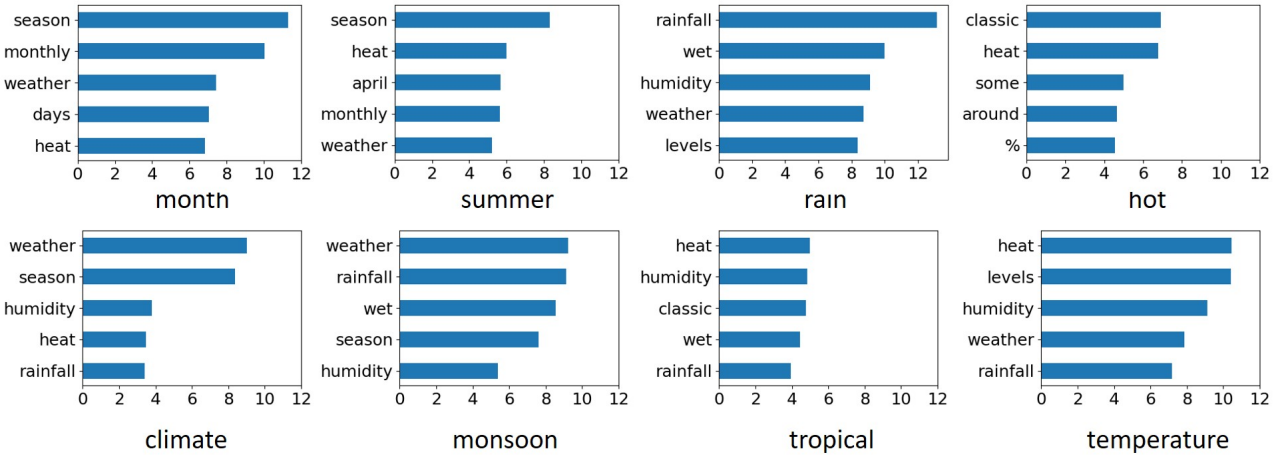


**Figure 4: The Top 5 contributing words to the expanded words of the second case in Figure 3. The X-axis are the words in the passage and Y-axis represents logit.**

## 6 CONCLUSION

In this work, we propose SparTerm to directly learn term-based sparse representation in the full vocabulary space. SparTerm learns a function to map the frequency-based and BoW representation to a sparse term importance distribution in the whole vocabulary space, which involves both term-weighting and expansion in the same framework. Experiments conducted on MSMARCO dataset show that SparTerm significantly outperforms previous sparse models based on the comparable size of PLMs, achieving state-of-the-art ranking performance among all sparse models. We conduct further empirical analysis about how the deep knowledge of PLMs can be transferred to the sparse method, which gives new insights for sparse representation learning. Empirical results show that SAPRT significantly increases the upper limit of sparse retrieval methods.

## ACKNOWLEDGEMENT

We thank Xin Jiang, Xiuqiang He, and Xiao Chen for the helpful discussions.

# REFERENCES


[1] D. Cheriton. 2019. From doc2query to docTTTTTquery.

[2] Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. *arXiv preprint arXiv:1910.10687* (2019).

[3] Zhuyun Dai and Jamie Callan. 2020. Context-Aware Document Term Weighting for Ad-Hoc Search. In *Proceedings of The Web Conference 2020*. 1897–1907.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[5] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[6] O. Khattab and M. Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020).

[7] John Lafferty and Chengxiang Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. 111–119.

[8] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300* (2019).

[9] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and M. Collins. 2020. Sparse, Dense, and Attentional Representations for Text Retrieval. *ArXiv* abs/2005.00181 (2020).

[10] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset *(CEUR Workshop Proceedings)*, Vol. 1773. CEUR-WS.org.

[11] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document Expansion by Query Prediction. *arXiv preprint arXiv:1904.08375* (2019).

[12] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).

[13] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, W. Li, and Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *ArXiv* abs/1910.10683 (2019).

[14] Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR¡ ¯94*. Springer, 232–241.

[15] K. Sparck-jones. 1972. A statistical interpretation of term specificity and its application in retrieval.

[16] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. 2005. Indri: A language model-based search engine for complex queries. In *Proceedings of the international conference on intelligent analysis*, Vol. 2. Citeseer, 2–6.

[17] Hamed Zamani, Mostafa Dehghani, W Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 497–506.

[18] Hugo Zaragoza, Nick Craswell, Michael J Taylor, Suchi Saria, and Stephen E Robertson. 2004. Microsoft Cambridge at TREC 13: Web and Hard Tracks.. In *TREC*, Vol. 4. 1–1.