applied sciences

MDPI

Article

Efficient Estimate of Low-Frequency Words' Embeddings Based on the Dictionary: A Case Study on Chinese

Xianwen Liao [1], Yongzhong Huang [1,*], Changfu Wei [2], Chenhao Zhang [1], Yongqing Deng [1] and Ke Yi [3]

[1] School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin 541004, China; liaoxianwen@mails.guet.edu.cn (X.L.); zhangchenhao2920@outlook.com (C.Z.); dengyq_397@163.com (Y.D.)
[2] School of Southeast Asian Studies, Guangxi University for Nationalities, Nanning 530006, China; weichfu@126.com
[3] College of Foreign Studies, Guilin University of Electronic Technology, Guilin 541004, China; yike19981025@126.com
* Correspondence: huangyongzhong@guet.edu.cn

Abstract: Obtaining high-quality embeddings of out-of-vocabularies (OOVs) and low-frequency words is a challenge in natural language processing (NLP). To efficiently estimate the embeddings of OOVs and low-frequency words, we propose a new method that uses the dictionary to estimate the embeddings of OOVs and low-frequency words. More specifically, the explanatory note of an entry in dictionaries accurately describes the semantics of the corresponding word. Naturally, we adopt the sentence representation model to extract the semantics of the explanatory note and regard the semantics as the embedding of the corresponding word. We design a new sentence representation model to encode sentences to extract the semantics from the explanatory notes of entries more efficiently. Based on the assumption that the higher quality of word embeddings will lead to better performance, we design an extrinsic experiment to evaluate the quality of low-frequency words' embeddings. The experimental results show that the embeddings of low-frequency words estimated by our proposed method have higher quality. In addition, both intrinsic and extrinsic experiments show that our proposed sentence representation model can represent the semantics of sentences well.

Keywords: natural language processing; word embedding; BERT; dictionary

Citation: Liao, X.; Huang, Y.; Wei, C.; Zhang, C.; Deng, Y.; Yi, K. Efficient Estimate of Low-Frequency Words' Embeddings Based on the Dictionary: A Case Study on Chinese. *Appl. Sci.* **2021**, *11*, 11018. https://doi.org/10.3390/app112211018

Academic Editor: Julian Szymanski

Received: 22 October 2021
Accepted: 19 November 2021
Published: 21 November 2021

1. Introduction

The embedding of a word corresponds to a point in the continuous multidimensional real number space, and the numerical embedding brings a lot of convenience to calculation. Word embeddings contain semantics and other information learned from the large-scale corpora. Recent works have demonstrated substantial gains on many natural language processing (NLP) tasks and benchmarks by pre-training on a large corpus of text followed by fine-tuning on a specific task [1,2]. Thus, many machine learning methods use pre-trained word embeddings as input and achieve better performance in many NLP tasks [3], such as the well-known text classification [4–6] and neural machine translation [7–9], among others.

One of the earliest studies on word representations dates back to 1986 and was conducted by Rumelhart, Hinton, and William [10]. In the following decades, many word embedding models based on the bag-of-words (BOW) language model (LM) and neural network LM have been proposed. These word embedding models include the well-known LDA [11], Word2Vec [12], Glove [13], ELMO [14], and BERT [1]. As soon as BERT was proposed, it outperformed the state-of-the-art methods on eleven NLP tasks. Usually, these word embedding models are trained using a huge corpus. However, for some low-resource languages, it is infeasible to construct a large corpus. When using a small corpus to estimate word embeddings, sparsity is a major problem. Sparsity leads

to out-of-vocabulary (OOV) everywhere. For some tasks that require word segmentation, the OOV phenomenon is more obvious. This is because word segmentation leads to more significant long-tail characteristics [15,16]. In addition, Zipf's law applies to most languages. This makes word embedding models unable to fully learn the semantics of OOVs and low-frequency words [16,17]. Therefore, accurately estimating the embeddings of OOVs and low-frequency words becomes the research motivation of this paper. We take Chinese as an example and use a dictionary to estimate the embeddings of those words. Different from English texts, there are no explicit delimiters such as whitespace to separate words in Chinese texts [15,18], just like the explanatory note in Figure 1. Therefore, Chinese word segmentation is important for some Chinese NLP tasks [15,18]. However, Chinese word segmentation will cause more serious sparsity problems, which makes the embeddings of OOVs and low-frequency words more difficult to be estimated [16].

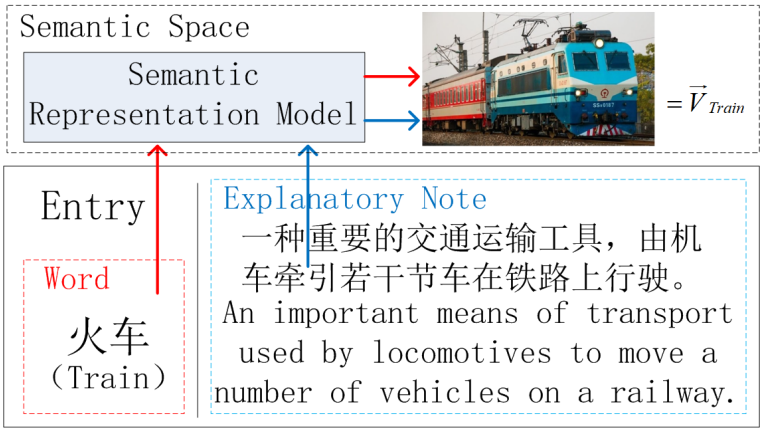

Figure 1. An entry's construct and the semantics relationship between the word and the explanatory note. A Chinese word usually contains multiple Chinese characters. Although "train" is not a low-frequency word in Chinese, we use it as an example for demonstration.

An entry in the dictionary contains a word and the corresponding explanatory note. They all point to the same point in the semantics space. As shown in Figure 1, the explanatory note usually contains rich information which explains the meaning of the corresponding word exactly. Inspired by this, we designed a semantics extractor to extract semantics from explanatory notes. We use the semantics representation produced by the extractor as the representation of low-frequency words. For high-frequency words, we still retain their word representations estimated by other word embedding models, such as Word2Vec. By combining the two types of word embedding estimation methods, we will obtain higher quality word representations that will be fine-tuned in downstream tasks. As the extrinsic experimental results in this paper show, the higher the quality of the word representation, the better the performance we will obtain. Our main contributions are as follows:

- We use the dictionary to estimate the embeddings of OOVs and low-frequency words. We also study the effects of low-frequency word embedding replacement rate on the performance of semantics match tasks.
- We propose a new sentence representation model which is different from the current mainstream LM, such as BERT [1], XLNet [19], and GPT [2,20].

2. Related Work

Our work mainly involves the estimation of OOVs and low-frequency words' embeddings and designing a sentence representation model. In this section, we introduce the related works of these two aspects.

Whether it is static word embedding models such as Word2Vec, Glove, and fasttext [12,13,21], or dynamic word embedding models such as BERT, ELMO, and GPT [1,2,14], they all extract features from a large number of samples to generate word representations. For OOVs that have never appeared and low-frequency words,

these models are unable to estimate their representation well [17]. Researchers have studied how to improve the estimate of OOVs and low-frequency words' representations. These methods mainly use the surface features of OOVs and their context to predict the meaning. Three types of embeddings (word, context clue, and subword embeddings) were jointly learned to enrich the OOVs' representations [22]. In [17], an OOV embedding prediction model named hierarchical context encoder (HiCE) was proposed to capture the semantics of context as well as morphological features. Recently, a mimicking approach has been found to be a promising solution to the OOV problem. In [23], an iterative mimicking framework that strikes a good balance between word-level and character-level representations of words was proposed to better capture the syntactic and semantic similarities. In [24], a method was proposed to estimate OOVs' embeddings by referring to pre-trained word embeddings for known words with similar surfaces to target OOVs. In [25], the embeddings of OOVs were determined by the spelling and the contexts in which they appear. The above-mentioned word embedding models that use morphology to infer the representations of OOVs are effective for English. However, they are not necessarily effective for Chinese, because many words with similar forms have very different meanings.

An explanatory note in an entry is usually a complete sentence. We naturally think of using the sentence representation model to extract the semantics from the explanatory note and treat it as the semantics of the corresponding word. In recent years, many sentence representation models have been proposed and widely used. Facebook AI's fasttext is a sentence representation model based continuous skip-gram model [12,21], which can estimate both word representations and sentence representations (https://github.com/facebookresearch/fastText, accessed on 5 March 2021). In [26], an unsupervised sentence embedding method (sent2vec) using compositional n-gram features was proposed to produce general-purpose sentence embeddings. Both fasttext and sent2vec are all BOW models, and we think that the BOW mechanism is just the simple combination of word embeddings. BERT is a landmark dynamic word embedding model. It learns sentence representations by performing two tasks: masked word prediction (MWP) and next sentence prediction (NSP) [1]. The embedding of token [CLS] in the last layer of BERT is considered as the representation of the input sentence. SBERT-WK is a sentence representation model based on BERT. It calculates the importance of words in sentences through subspace analysis and then weights word embeddings to generate sentence representations [27]. In [28], the framework of neural machine translation (LASER) was adopted to jointly learn sentence representations across different languages. BERT uses a bidirectional self-attention encoder (the transformer) to encode sentences, and LASER uses a BiLSTM. In addition, there are more studies on sentence representations [29–31].

There are many publicly available sentence representation models. [1,21,26,27,32]. However, so far, there is no sentence representation model without any flaws. BERT only uses the embedding of [CLS] in the last layer of BERT to represent the input sequence [1]. The embedding of [CLS] is mainly learned from NSP. However, a recent study shows that NSP does not contribute much to the sentence representation learning [33]. SBERT-WK can make use of the existing semantics in BERT as much as possible, but it cannot increase the semantics in BERT. LASER is a universal multilingual sentence representation model covering more than 100 languages [32], and it uses BiLSTM to encode sentences. We believe that there are some limitations to constructing a large and high-quality parallel corpus, and the encoding ability of BiLSTM is also inferior to Transformer. sent2vec and fasttext are BOW LMs, and they both use the n-gram features to represent the semantics of sentences [21,26]. Thus, in this article, we propose a new sentence representation model from a new perspective.

3. Lessons Learned from an Infeasible Heuristic Model

In this section, we introduce lessons learned from an infeasible heuristic model. The lessons guide us to construct our new sentence representation model.

As Figure 1 shows, an entry consists of a word and its explanatory note. A word and its explanatory note all point to the same point in the semantics space. The natural idea is to construct two different encoders to encode the word and its explanatory note. The two encoders shown in Figure 2 are trained with the goal of minimizing the difference of the two outputted semantics vectors. WE and ENE do not share parameters, and they both use a BiLSTM or BiGRU to encode the token sequence. The detail of the encoder can be seen in [34].

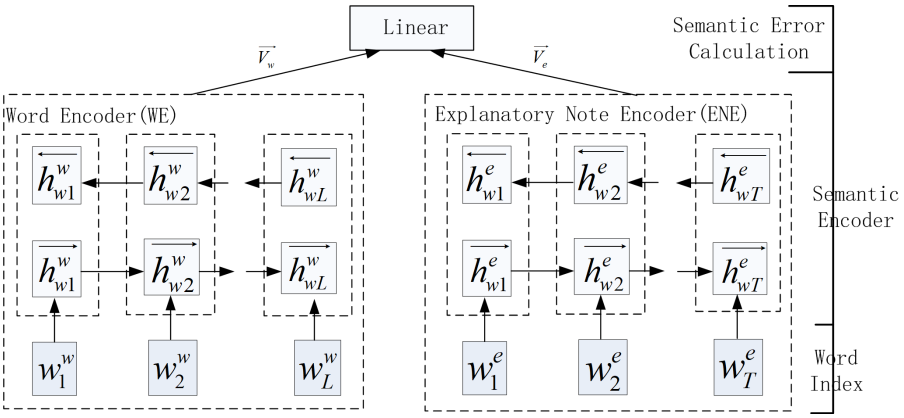

Figure 2. An infeasible semantics learning model composed of WE and ENE. The model aims to extract the semantics of words from the explanatory notes but fails.

Suppose we have an entry $W_1^w W_2^w \dots W_L^w : W_1^e W_2^e \dots W_T^e$. $W_1^w W_2^w \dots W_L^w$ and $W_1^e W_2^e \dots W_T^e$ which represent a word and its explanatory note in an entry. We use $\overrightarrow{V_w}$ and $\overrightarrow{V_e}$ to denote the semantics vectors of the word and the explanatory note. We use the euclidean distance (or the cosine distance) to measure the semantics difference between the two vectors. To make the semantics difference as small as possible, we design the objective function defined by Equation (1) and minimize it to train WE and ENE shown in Figure 2.

$$\text{Loss} = \sum_{i \in D} ed(\overrightarrow{V_w^i}, \overrightarrow{V_e^i}). \tag{1}$$

So far, everything seems to be going according to our expectations. Unfortunately though, no matter how we jointly train WE and ENE, the parameters being trained always converge to **0**. The output of WE and ENE also tend to $\overrightarrow{0}$, that is, the semantics vector we finally obtain tends to $\overrightarrow{0}$. Why? Because **0** is one of the feasible solutions of the model, and 0 is the minimum loss of the objective function defined by Equation (1). With the effective search of optimization algorithm (we use Adam [35] to optimize the model), the loss of the objective function tends to 0 finally, and the parameters being trained also tend to **0**. Therefore, we can conclude that we cannot approximate an implicit objective that varies with optimization parameters because such implicit objectives make the loss function of the model have zero solutions, and when the loss function is equal to 0, all parameters are zero. This is why pre-trained LMs such as Word2Vec, BERT, XLNet, and GPT [1,2,12,19] all regard words in the vocabulary as prediction targets (the predicted words are fixed and will not vary with the trainable parameters). Based on this principle, we design a new sentence representation model.

4. The Proposed Sentence Representation Model

As Figure 1 shows, an entry consists of a word and the corresponding explanatory note. Usually, an explanatory note is a complete sentence, so we extract the semantics of the explanatory note and treat it as the representation of the corresponding word.

We use $s_{1:m} = T_1 \dots T_i \dots T_m$ to represent a sentence with length m. The sentence representation of $s_{(1:m)}$ is denoted by $SEMTS(s_{(1:m)})$. We assume that the more tokens a

sentence contains, the more semantics it conveys. Let us consider two token sequences, $s_{(1:k)}$ and $s_{(1:k+1)}$. $s_{(1:k+1)}$ has one more token T_{k+1} than $s_{(1:k)}$. According to the hypothesis, $s_{(1:k+1)}$ contains more semantics than $s_{(1:k)}$. The added semantics of $s_{(1:k+1)}$ than $s_{(1:k)}$ is mainly caused by T_{k+1}, so we can use the added semantics to predict T_{k+1}, that is,

$$SEMTS(s_{(1:k+1)}) - SEMTS(s_{(1:k)}) \xrightarrow{\text{predict}} T_{k+1}. \tag{2}$$

In Equation (2), the sentence representation $SEMTS(s_{(1:k+1)})$ is calculated by the self-attention mechanism. The self-attention mechanism in our sentence representation model shown in Figure 3 is slightly different from the traditional self-attention mechanism [1]. Suppose that $Z = Z_1 Z_2 \cdots Z_m$ is the output of the encoder when the input token sequence is $s_{(1:m)}$. Z_i is the encoding of T_i. The calculation process of the sentence representation of $s_{(1:m)}$ is shown in Figure 3.

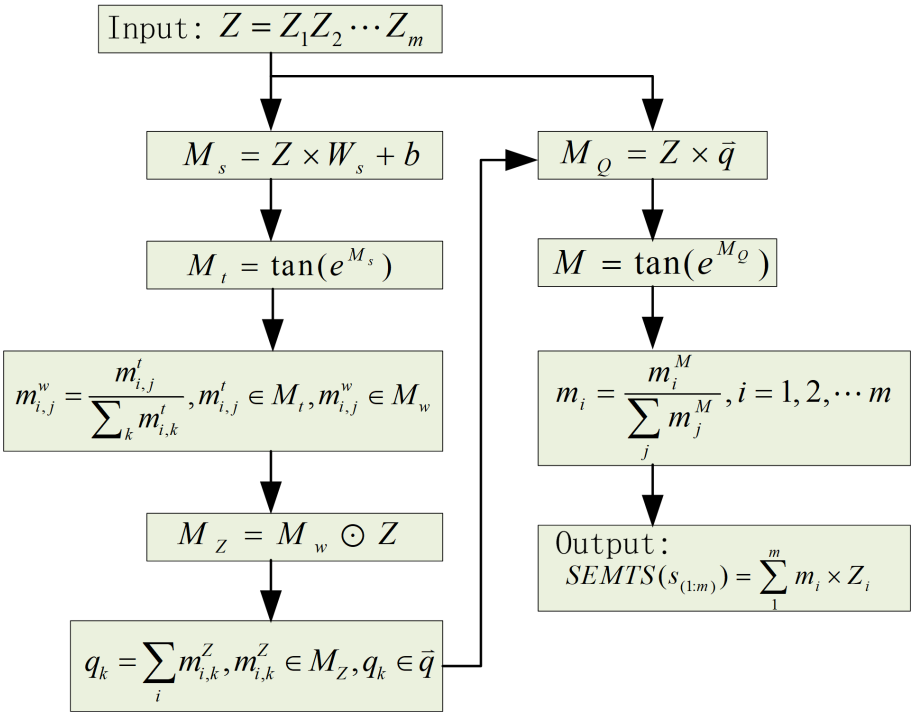

Figure 3. The calculation process of the sentence representation denoted by $SEMTS(s_{(1:m)})$.

In Figure 3, $\vec{q}$, W_s, and b represent the query vector, the shape transformation matrix, and the bias, respectively. The e, tan, and $\odot$ operations are all element-wise.

To make full use of the strong encoding ability of BERT, we use BERT as the backbone of our sentence representation model. Another benefit of building and fine-tuning the model based on BERT is that it can save a lot of computation power. Thus, we add our semantics computing module to BERT as a sub-task, and we call our semantics representation model SEMTS-BERT. The architecture of SEMTS-BERT is shown in Figure 4. NSP and MWP are two sub-tasks of the original BERT [1]. Final Word Prediction (FWP) sub-task corresponds to our semantics computing model, and its structure is shown in Figure 5. The loss of FWP sub-task is defined as

$$\text{Loss}_{\text{fwp}} = -\frac{1}{N}\sum_1^N p(T_{k+1}), \tag{3}$$

where $p(T_{k+1})$ is the probability of T_{k+1} defined in Equation (2) when using the added semantics to predict T_{k+1}. N is the number of predicted words in a batch. $p(T_{k+1})$ is calculated by FWP, shown in Figure 5. When we minimize the objective, the loss of FWP defined in this way makes the probability of predicted words as large as possible.

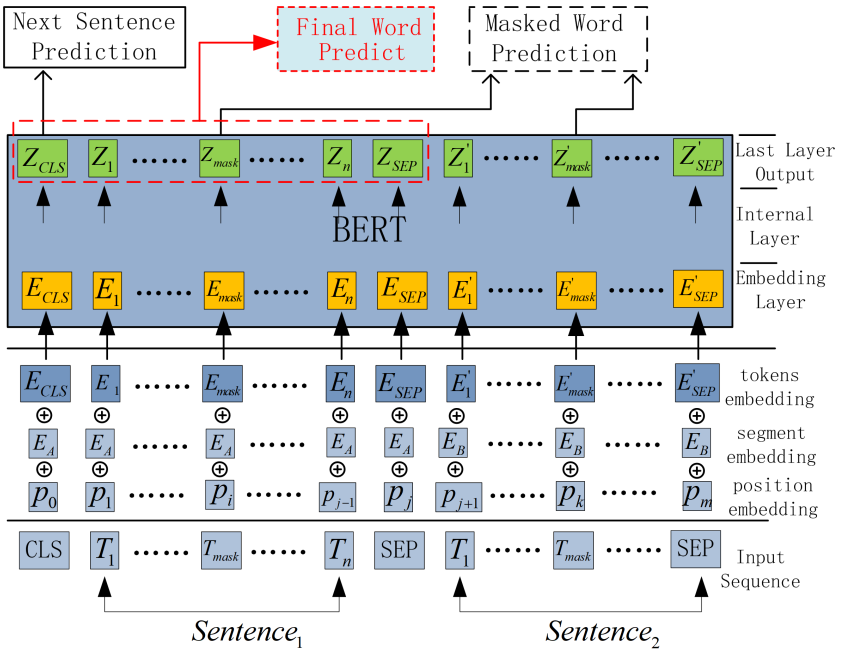

Figure 4. The architecture of SEMTS-BERT. FWP is added to BERT as a sub-task.

The total loss of SEMTS-BERT is the sum of Loss_{fwp}, Loss_{nsp}, and Loss_{mwp}, that is,

$$\text{Loss} = \text{Loss}_{fwp} + \text{Loss}_{nsp} + \text{Loss}_{mwp}\,. \tag{4}$$

The detail of Loss_{nsp} and Loss_{mwp} can be seen in [1].

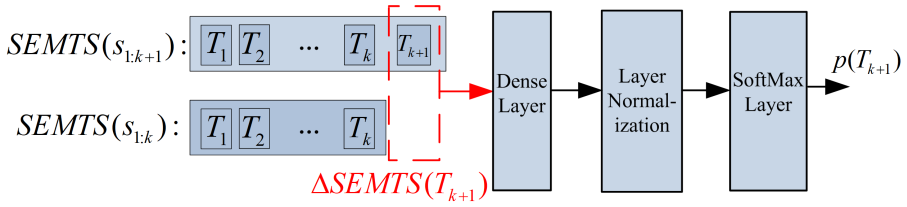

Figure 5. The architecture of FWP. We follow the practical experience of BERT and add a dense layer before the softmax layer. A dense layer is a full-connected layer. The function of the softmax layer is to calculate the probability distribution of the output, and the probability of the output is usually defined as: $p_{y_k} = \frac{e^{y_k}}{\sum_j e^{y_j}}$.

5. Experiment

In this section, we choose Chinese as a study case and perform two types of experiments: intrinsic evaluation and extrinsic evaluation [17]. The intrinsic evaluation is designed to evaluate the effectiveness of our proposed SEMTS-BERT. It includes three tasks: a probing task [36], a text classification task, and a Natural Language Inference (NLI) task. The extrinsic evaluation is designed to verify the quality of OOVs and low-frequency words' embeddings. In the extrinsic experiment, we replace OOVs and low-frequency words' embeddings in two downstream tasks: sentence semantic equivalence identification (SSEI) and question matching (QM) [37,38]. We also evaluate the quality of low-frequency words' embeddings by investigating the relative distance between similar words.

5.1. Experimental Settings

As shown in Figure 4, our model is composed of FWP module and BERT. We initialize our model with the Chinese 12-layer, 768-hidden, 12-head, 110M parameter BERT-Base model (https://github.com/google-research/bert, accessed on 10 March 2021) and train it with a dataset derived from texts (250M bytes) downloaded from Wikipedia (https:

//dumps.wikimedia.org/zhwiki/, accessed on 1 June 2020). We take a Chinese sentence "哲学研究的是基础的问题。" ("Philosophy studies basic issues") as an example to illustrate the construction of the dataset. We use the Adam optimizer (the initial learning-rate and the warm-up steps are set to 2×10^{-5} and 12,000) to train SEMTS-BERT 2 epochs [35]. The batch size is 2 and the maximum sequence length is set to 128. As shown in Figure 6, a sentence can derive many examples. We can obtain nearly 200 examples when the batch size and the maximum sequence length are set to 2 and 128. When SEMTS-BERT has been trained, we use the process shown in Figure 3 to calculate sentences' representation. For an entry in dictionaries, we input the explanatory note into SEMTS-BERT, and the output is the representation of the corresponding word.

$S_{1:k+1}$	$S_{1:k}$	T_{k+1}
[CLS]哲学研究的是基础的问题。[SEP]	[CLS]哲学研究的是基础的问题。	[SEP]
[CLS]哲学研究的是基础的问题。	[CLS]哲学研究的是基础的问题	。
[CLS]哲学研究的是基础的问题	[CLS]哲学研究的是基础的问	题
[CLS]哲学研究的是基础的问	[CLS]哲学研究的是基础的	问
[CLS]哲学研究的是基础的	[CLS]哲学研究的是基础	的
[CLS]哲学研究的是基础	[CLS]哲学研究的是基	础
[CLS]哲学研究的是基	[CLS]哲学研究的是	基
[CLS]哲学研究的是	[CLS]哲学研究的	是
[CLS]哲学研究的	[CLS]哲学研究	的
[CLS]哲学研究	[CLS]哲学研	究
[CLS]哲学研	[CLS]哲学	研
[CLS]哲学	[CLS]哲	学
[CLS]哲	[CLS]	哲

Figure 6. The construction of the dataset used to train SEMTS-BERT. The symbols $s_{1:k+1}$, $s_{1:k}$, and T_{k+1} are defined in Figure 5. [CLS] and [SEP] are two special characters used to enclose sentences.

5.2. Baselines

We choose the following models as baselines to evaluate the performance of SEMTS-BERT. The performance of sentence representation models directly determines the quality of low-frequency words' embeddings.

- BERT: In addition to estimating dynamic word embedding, BERT can also be used to calculate the embedding of a sentence (the encoding of [CLS] in the last layer is treated as the sentence representation) [1].
- fasttext https://fasttext.cc/, accessed on 7 March 2021: fasttext is a pre-training BOW model for 157 different languages. It is a famous library for estimating both words and sentences [21].
- sent2vec http://github.com/epfml/sent2vec, accessed on 7 March 2021: sent2vec is an efficient unsupervised BOW model, and it uses word embeddings and n-gram embeddings to estimate sentence representations [26].
- LASER https://github.com/facebookresearch/LASER, accessed on 8 March 2021: LASER is a multilingual sentence representation model. It adopts BiLSTM as an encoder which was trained on a parallel corpus that covers 93 languages [28].
- SBERT-WK: SBERT-WK is a sentence representation model based on BERT. It calculates the importance of words in sentences through subspace analysis and then weights the word embeddings to obtain sentence representations [27].

5.3. Intrinsic Evaluation 1: Evaluate SEMTS-BERT on Probing Tasks

Probing tasks are designed to evaluate the performance of models on capturing the simple linguistic properties of sentences [39]. We use the Chinese CoNLL2017 (http://universaldependencies.org/conll17/data.html, accessed on 7 June 2020) for this evaluation. The dataset is uneven, and its statistics are shown in Table 1. We adopt some sub-tasks defined in [36,39] in our evaluation. They are:

- Sentence_Len (sentence length): We divide sentences into two classes: class 0 (its length is shorter than the average length) and class 1 (its length is longer than the

average length). In this test, the classifier is trained to tell whether a sentence belongs to class 0 or 1.

- Voice: The goal of this binary classification task is to test how well the model can distinguish the active or passive voice of a sentence. In the case of complex sentences, only the voice of the main clause is detected.
- SubjNum: In this binary classification task, sentences are classified by the grammatical number of nominal subjects of main predicates. There are two classes: sing and plur.
- BShift: In the BShift dataset, we exchange the positions of two adjacent words in sentence. In this binary classification task, models must distinguish intact sentences from sentences whose word order is illegal.

Table 1. The detail of the Chinese probing dataset derived from CoNLL2017.

Probing Task	Class	Train Set	Development Set	Test Set
Sentence_Len	Long Sentence	1571	209	195
	Short Sentence	2419	288	305
Voice	Passive Voice	311	35	37
	Active Voice	3689	461	462
Bshift	Shift	3993	498	499
	Non-shift	3993	498	499
SubjNum	Plur	115	21	20
	Sing	3796	466	471

In this test, we fit a Multi-Layer Perception (MLP) with one hidden layer on the top of the sentence representation model to perform the classification [40]. The architecture of MLP is illustrated in Figure 7.

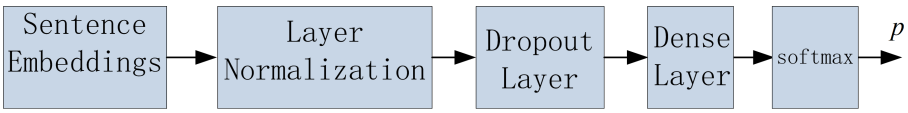

Figure 7. The architecture of MLP for probing tasks.

5.3.1. Experimental Results

The experimental results of probing tasks are shown in Table 2. We use accuracy to express the performance since $P = R = F$ in classification. P, R, and F represent precision, recall, and F-measure: $P = \frac{true\ positives}{predicted\ as\ positives}$, $R = \frac{true\ positives}{actual\ positives}$, $F = \frac{2PR}{P+R}$. We use the mean and standard deviation (number in brackets) to express the performance of models. $BERT_{cls}$ means that the encoding of the token [CLS] is treated as the representation of the input sentence. $BERT_{max}$ means that the max-pooling of the encodings in BERT's last layer is treated as the representation of the input sentence, and $BERT_{mean}$ means that the mean-pooling of the encodings in BERT's last layer is treated as the representation of the input sentence. $fasttext_{max}$ and $fasttext_{mean}$ are the same as $BERT_{max}$ and $BERT_{mean}$. $fasttext_{cls}$ represents the native sentence representation model of fasttext. We input the embeddings of sentences into the classifier described in Figure 7 to evaluate each sentence representation model.

Table 2. The experimental results of different models in probing tasks.

Model	Probing Tasks				Avg.
	Sentence_Len	Bshift	Voice	SubjNum	
$BERT_{cls}$	0.431(0.129)	**0.871(0.188)**	0.927(0.017)	**0.983(0.007)**	0.803(0.085)
$BERT_{max}$	0.561(0.136)	0.779(0.110)	0.972(0.021)	0.829(0.125)	0.785(0.098)
$BERT_{mean}$	0.516(0.159)	0.687(0.200)	0.981(0.008)	0.825(0.253)	0.752(0.155)
$fasttext_{cls}$	0.825(0.005)	0.505(0.002)	0.951(0.002)	0.957(0.002)	0.810(0.003)
$fasttext_{max}$	0.781(0.006)	0.497(0.002)	0.926(0.001)	0.963(0.002)	0.792(0.003)
$fasttext_{mean}$	0.804(0.006)	0.513(0.002)	0.950(0.001)	0.963(0.003)	0.808(0.003)
sent2vec	0.861(0.005)	0.509(0.001)	**0.983(0.003)**	0.965(0.003)	0.830(0.003)
LASER	0.938(0.005)	0.563(0.006)	0.950(0.002)	0.967(0.002)	0.855(0.004)
SBERT-WK	0.936(0.006)	0.731(0.004)	0.938(0.005)	0.962(0.001)	0.892(0.004)
SEMTS-BERT	**0.959(0.001)**	0.740(0.005)	0.956(0.003)	0.972(0.002)	**0.907(0.003)**

From Table 2, we can see that $BERT_{cls}$ performs best on Bshift and subjNum but worst on Sentence_Len. SEMTS-BERT and *sent2vec* perform best on Sentence_Len and Voice. In the Voice sub-task, all models achieve good performance. In Bshift, all models except $BERT_{cls}$ do not perform well. Among all the models, the standard deviation of *BERT* is the largest, which shows that its training results are unstable and tend to fall into the local extremum easily during the optimization process. We try to reduce the number of neurons in the hidden layer to reduce *BERT*'s standard deviation, but doing so will reduce its overall performance. Compared with the native sentence representation model, max and mean pooling operations sometimes achieve better performance.

5.3.2. Analysis and Conclusion

SEMTS-BERT has the best overall performance, followed by SBERT-WK and *LASER*. Compared with *BERT*, *fasttext*, and *sent2vec*, the performance of *LASER* has been proven to be the best in probing tasks [36]. This shows that SEMTS-BERT has good performance on probing tasks. Due to the self-attention mechanism, SEMTS-BERT cannot retain the positional relationship of tokens well, but its performance on Bshift is still better than *fasttext*, *sent2vec*, and *LASER*. $BERT_{cls}$ performs best on Bshift because it can obtain the token position information from residual connections between layers and local features from the masked LM [1]. From Bshift, we can draw a conclusion that the weighted operation cannot filter all the positional information. *fasttext* and *sent2vec* are based on the BOW LM, so it is easy to understand that they do not perform well on this task. However, LASER uses BiLSTM to encode token sequences, and it should be sensitive to the word order, but the experimental results are not the same as our expectation.

5.4. Intrinsic Evaluation 2: Evaluate SEMTS-BERT on Text Classification

Text classification is a popular task used to evaluate the performance of models in NLP [5]. In this section, we evaluate the performances of different sentence representation models on three Chinese text classification datasets.

- Thucnews: A high-quality 14-category text classification dataset containing 0.74 M news articles collected from Sina News: https://news.sina.com.cn/, accessed on 1 May 2021. The dataset is provided by the NLP Laboratory of Tsinghua University: http://thuctc.thunlp.org/, accessed on 1 May 2021.
- Fudan Dataset (Fudan): Fudan dataset: http://www.nlpir.org/?action-viewnews-itemid-103, accessed on 1 May 2021, is a 20-category text classification dataset that contains 9833 test documents and 9804 training documents. It is an uneven dataset since the number of documents of each category varies greatly.
- TouTiao Dataset (TouTiao): TouTiao dataset is a 15-category short text classification dataset. Each example contains only a news headline and a subheading. It is noisy and collected from TouTiao news website: www.toutiao.com, accessed on 1 May 2021.

We first split paragraphs into sentences and the calculate the representations of sentences. We later use a BiLSTM encoder to encode the sentence representation sequence. Finally, the outputs of the encoder (document representations) are inputted into the softmax layer to perform text classification. The architecture of the text classifier is shown in Figure 8. We use the Adam optimizer [35] to train each model 50 epochs with early stop strategy on every dataset.

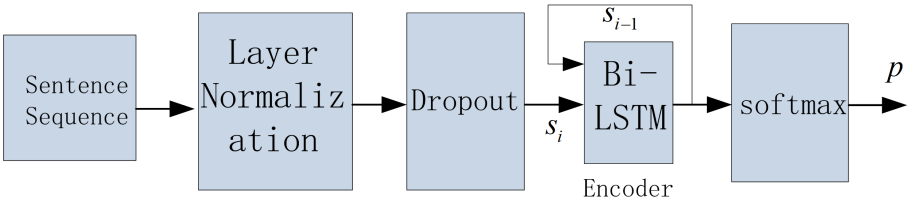

Figure 8. The network architecture of the text classifier.

5.4.1. Experimental Results

We list the experimental results of this evaluation in Table 3. *cls*, *max*, and *mean* have the same meaning as those in Table 2. We can easily see that our SEMTS-BERT performs best. The variance of BERT and fasttext on Toutiao and Thucnews is very large, which shows that the text classification results of these two models are unstable. Due to the vagueness and ambiguity of some examples in Toutiao, the classification accuracy of all models on Toutiao is not high, and the results are not stable.

Table 3. Experimental results of text classification.

Model	Datasets		
	Thucnews	**Toutiao**	**Fudan**
$BERT_{cls}$	0.874(0.007)	0.637(0.015)	0.925(0.009)
$BERT_{max}$	0.420(0.023)	0.492(0.020)	0.870(0.016)
$BERT_{mean}$	0.901(0.011)	0.735(0.006)	0.932(0.006)
$fasttext_{cls}$	0.956(0.005)	0.868(0.012)	0.962(0.009)
$fasttext_{max}$	0.934(0.015)	0.835(0.017)	0.935(0.007)
$fasttext_{mean}$	0.956(0.007)	0.878(0.015)	0.956(0.010)
sent2vec	0.945(0.006)	0.856(0.013)	0.963(0.005)
LASER	0.948(0.005)	0.835(0.008)	0.959(0.005)
SBERT-WK	0.955(0.002)	0.855(0.005)	0.971(0.003)
SEMTS-BERT	**0.963(0.003)**	**0.880(0.013)**	**0.973(0.002)**

5.4.2. Analysis and Conclusions

Our model obtains the best performance on this test. This shows that our sentence encoding mechanism can represent the semantics of sentences well. The overall performance of BERT model is worst, which shows that BERT needs further fine-tuning to obtain better performance in downstream tasks (sentence embeddings are fixed in this text). $fasttext_{cls}$ has achieved second only to us, and its performance has surpassed LASER. This shows that in text classification, a simple BOW model can also perform well. Compared with BERT, SBERT-WK achieves better performance, which indicates that the sentence representation obtained by weighting the embeddings in each layer has better performance than using only the output of the last layer as the sentence representation.

5.5. Intrinsic Evaluation 3: Evaluate SEMTS-BERT on Natural Language Inference (NLI)

In NLI task, a classifier is trained to determine whether one sentence entails, contradicts another sentence, or neither [41]. We use the Chinese XNLI corpus for this test [41]. The Chinese XNLI dataset contains 2312 and 4666 sentence pairs in its development set and

test set. We randomly re-divide them into train, development, and test sets. The classifier used in this test is shown in Figure 9. $\oplus$ represents the concatenation of the two vectors.

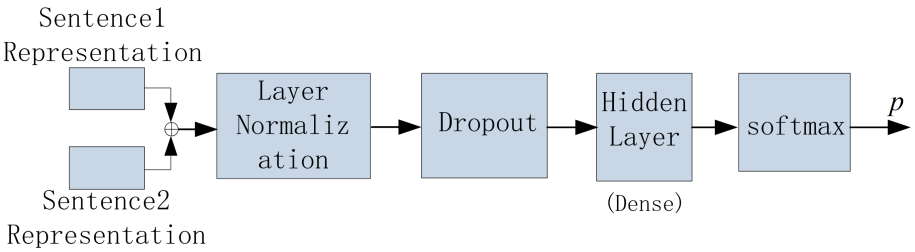

Figure 9. The architecture of the NLI classifier.

5.5.1. Experimental Results

The experimental results of the Chinese NLI are shown in Table 4. The meanings of *cls*, *max*, and *mean* are the same as those in Table 2. LASER performs the best, followed by our model. The standard deviation of *BERT* is very large, which shows that the classifier can easily fall into a local extremum during the training process. The *cls* models of *BERT* and *fasttext* perform better than *max* and *mean* models. This shows that the native sentence models are more suitable to the NLI task than the pooling models. The *mean* model performs better than the *max* model. The same conclusion is obtained in [36].

Table 4. The experimental results of the Chinese NLI. *acc* and *std*. represent classification accuracy and standard deviation.

Model	*acc*(*std*.)
$BERT_{cls}$	0.569(0.166)
$BERT_{max}$	0.380(0.179)
$BERT_{mean}$	0.465(0.188)
$fasttext_{cls}$	0.552(0.006)
$fasttext_{max}$	0.503(0.005)
$fasttext_{mean}$	0.549(0.007)
*sent*2*vec*	0.558(0.007)
LASER	**0.645(0.008)**
SBERT-WK	0.545(0.101)
SEMTS-BERT	0.604(0.005)

5.5.2. Analysis and Conclusion

BiLSTM can encode sequences well [42]. LASER uses a BiLSTM to encode token sequences, and it is trained on a large multilingual parallel corpus [32]. Therefore, it is understandable that it performs best, and the same conclusion has also been drawn in [36]. BERT uses a Transformer-based bidirectional encoder to encode the sequence [1]. In this test, the performance of BERT is not as good as LASER. This is because the sentence embeddings are fixed, and BERT can not benefit from the larger, more expressive pre-training representations [1]. However, our sentence representation model can improve this situation. *sent*2*vec* and *fasttext* have the worst performance. They both use a BOW LM to represent the semantics of sentences [21,26]. The sequence encoding ability of BOW is obviously inferior to BiLSTM and Transformer [36]. Surprisingly, the performance of SBERT-WK is not as good as $fasttext_{cls}$ and *sent*2*vec*. This shows that the complex weighting operation in SBERT-WK cannot improve the NLI performance of BERT. Because we only use simple classifier and the sentence representations are fixed, the optimal performance in this test is not high. However, this is enough for us to compare the semantics representation abilities of different models.

5.6. Extrinsic Evaluation 1: Evaluate the Quality of Low-Frequency Words' Embeddings by the Relative Distance

In this evaluation, we assume that in a set of words with similar meanings, if their embeddings are more concentrated, the quality of their embeddings will be better. We choose nine Chinese low-frequency words with similar meanings for this evaluation. They are 砂仁 (Fructus Amomi), 腽肭脐 (Testiset Penis Phocae), 枳壳 (Fructus Aurantii), 枳实 (Fructus), 紫河车 (Placenta Hominis), 阿胶 (Donkey-Hide Gelatin), 白药 (Baiyao, a white medicinal powder for treating hemorrhage, wounds, bruises, etc.), 膏药 (Plaster), 槐豆 (Locust Bean). These words are the names of some traditional Chinese medicines.

We use Word2Vec (trained on the Chinese text corpus downloaded from Wikidata) to estimate the embeddings of these words. We use SEMTS-BERT, SBERT-WK, and LASER to calculate the embedding of the explanatory note and regard the embedding as the embedding of the corresponding word. For example, the explanatory note of 砂仁 in the dictionary is "阳春砂或缩砂密的种子，入中药，有健胃、化滞、消食等作用". We input this explanatory note into a sentence representation model, and the produced embedding is treated as the embedding of 砂仁 .

The relative distances between the nine Chinese low-frequency words are shown in Figure 10. The embeddings estimated by our model have a smaller relative distance. Therefore, we think that our method can produce higher-quality low-frequency word embeddings.

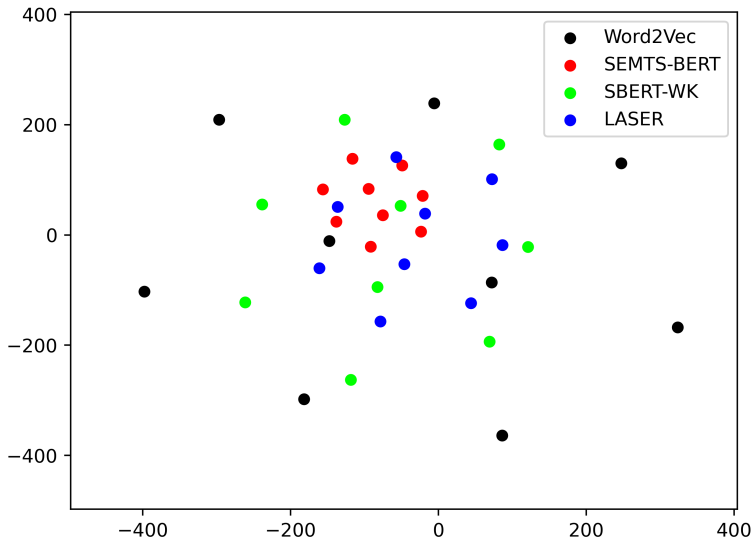

Figure 10. The relative distance between the Chinese low-frequency words. We use t-SNE (initialized by PCA) for dimensionality reduction so that the high-dimensional embeddings can be shown on a two-dimensional plane [43].

5.7. Extrinsic Evaluation 2: Evaluate the Quality of Low-Frequency Words' Embeddings on Downstream Tasks

We assume that the higher the quality of word embeddings, the better the performance we will obtain. Based on this assumption, we indirectly evaluate the quality of word embeddings through the performance of specific tasks. We adopt SSEI (sentence semantic equivalence identification) and QM (question matching) to evaluate the performance improvement caused by replacing the embeddings of low-frequency words [37,38]. By replacing the embeddings of OOVs and low-frequency words in train, development, and test sets, we can evaluate whether our proposed low-frequency word embedding estimation method can improve the performance of SSEI and QM as well as how much the performance has been improved. From the improvement, we can determine whether the quality of low-frequency words' embeddings has been improved.

5.7.1. Experimental Design

We use the Chinese dictionary named XIANDAI HANYU CIDIAN and choose two NLP tasks (SSEI and QM) for this evaluation. SSEI is a fundamental task of NLP in question answering (QA), automatic customer service, and chatbots. In customer service systems, two questions are defined as semantically equivalent if they convey the same intent or they could be answered by the same answer. Because of rich expressions in natural languages, SSEI is a challenging NLP task [37]. QM is also a fundamental task of QA, which is usually recognized as a semantic matching task, sometimes a paraphrase identification task. The goal of QM is to search questions that have similar intent as the input question from an existing database [38].

Without replacing the OOVs and low-frequency words' embeddings in train, development, and test sets, we first evaluate the performance of the baseline sentence matcher on the two datasets. We later replace low-frequency words' embeddings in the same dataset, and we evaluate the performance of the baseline sentence matcher again. By comparing the two results, we can obtain the performance improvement and determine whether our proposed method improves the quality of low-frequency words' embeddings. We use Word2Vec (https://github.com/RaRe-Technologies/gensim, accessed on 10 June 2021) to estimate word embeddings on a large Chinese corpus downloaded from Wikidata [12]. We use SEMTS-BERT, SBERT-WK, and LASER to estimate OOVs and low-frequency words' embeddings. When we input the explanatory note of an entry (suhc as the one shown in Figure 1) into a sentence representation model, the outputted sentence representation is considered as the embedding of the corresponding word. We choose LASER and SBERT-WK for the comparison because they have been proven high-performance [27,36].

5.7.2. Datasets

We use BQ [37] and LCQMC [38] datasets for this evaluation. Each example in BQ and LCQMC contains a sentence pair and a label. The label indicates whether the two sentences in sentence pairs match. The train, development, and test sets of BQ contain 100k, 10k, and 10k examples, respectively, and the train, development, and test sets of LCQMC contain 238k, 8.8k, and 12.5k examples, respectively. We use jieba (https://pypi.python.org/pypi/jieba, accessed on 1 July 2021) to perform Chinese word segmentation. The distributions of low-frequency words of the two datasets are shown in Table 5.

Table 5. The distributions of low-frequency words of BQ and LCQMC datasets.

Word Frequency	Percentage in BQ (%)	Percentage in LCQMC (%)
≤200	1.95	3.25
≤500	2.30	5.85
≤700	2.89	6.86
≤1000	5.33	7.98
≤3000	8.25	11.19
≤8000	11.00	13.36
≤20,000	12.99	14.75
≤50,000	13.79	15.75

5.7.3. Baseline and Parameter Settings

We choose BiLSTM, Text-CNN, DCNN, DIIN, BiMPM, and other machine learning models as sentence matcher baselines [44–48]. BiMPM is a character+word model [46], and it obtains the best results on BQ and LCQMC. The embeddings of characters are tuned, and the embeddings of words can be dynamic (tuned) or static (not to be tuned) in the evaluation.

5.7.4. Experimental Results

The comparisons between our model and other models on BQ and LCQMC are shown in Tables 6 and 7, Appendices A and B. "c" and "w" in the *Emb* column represent the character-based and the word-based model. "Acc." represents the classification accuracy. "+st." and "+dy." denote that word embeddings are fixed and to be tuned during the training. "+LASER", "+SB-WK", and "+SEMTS" mean that the embeddings used to replace the original embeddings of low-frequency words are calculated by LASER, SBERT-WK, and SEMTS-BERT, respectively. On both BQ and LCQMC, we all use the word-based BiMPM. On the BQ dataset, we obtain the similar performance to the benchmark obtained by a character-based model [37]. Generally, the performance of character-based model is better than the performance of word-based model on small Chinese datasets due to sparsity [16]. On the LCQMC dataset, we achieve better performance than the benchmark.

From Tables 6 and 7, Appendices A and B, we can draw a conclusion that we achieve better performance when we replace the low-frequency words' embeddings on the larger LCQMC dataset. The performance of the word-based model exceeds that of the character-based model after performing the replacement. This shows that we can obtain higher-quality word embeddings through the proposed method when the dataset is large. On the smaller BQ dataset, the replacement also promotes the word-based model. Sometimes, the performance of word-based models exceeds that of the character-based model after performing the replacement. In summary, the performance improvement indicates that our method can provide higher-quality low-frequency words' embeddings.

Table 6. The comparison between our method and other models on BQ.

Model	Emb	P	R	F	Acc.
TF-IDF	c	64.68	60.94	62.75	63.83
Text-CNN [44]	c	67.77	70.64	69.17	68.52
Text-CNN [44]	w	69.61	67.00	68.28	67.56
Text-CNN [44]+st.+LASER	w	68.96	68.38	68.67	68.30
Text-CNN [44]+dy.+LASER	w	68.63	67.09	67.85	68.87
Text-CNN [44]+st.+SB-BK	w	67.82	69.42	68.61	68.22
Text-CNN [44]+dy.+SB-BK	w	68.38	67.70	68.04	67.58
Text-CNN [44]+st.+SEMTS	w	67.91	70.15	69.01	68.49
Text-CNN [44]+dy.+SEMTS	w	68.65	69.01	68.83	67.37
BiLSTM [48]	c	75.04	70.46	72.68	73.51
BiLSTM [48]	w	74.79	68.52	71.52	71.06
BiLSTM [48]+st.+LASER	w	72.68	74.28	73.47	73.55
BiLSTM [48]+dy.+LASER	w	73.01	73.39	73.20	72.73
BiLSTM [48]+st.+SB-BK	w	72.85	72.73	72.79	73.28
BiLSTM [48]+dy.+SB-BK	w	72.63	73.09	72.86	72.66
BiLSTM [48]+st.+SEMTS	w	73.94	72.83	73.38	73.45
BiLSTM [48]+dy.+SEMTS	w	72.86	73.10	72.98	72.89
DIIN [45]	c	81.58	81.14	81.36	81.41
DIIN [45]	w	81.71	79.23	80.45	80.78
DIIN [45]+st.+LASER	w	81.50	81.16	81.33	81.27
DIIN [45]+dy.+LASER	w	80.97	81.33	81.15	80.85
DIIN [45]+st.+SB-BK	w	81.02	81.50	81.26	81.29
DIIN [45]+dy.+SB-BK	w	81.91	79.86	80.87	80.75
DIIN [45]+st.+SEMTS	w	81.07	81.63	81.35	81.39
DIIN [45]+dy.+SEMTS	w	81.56	80.11	80.83	80.91

Table 6. *Cont.*

Model	Emb	P	R	F	Acc.
BiMPM [46]	c	82.28	81.18	81.73	**81.85**
BiMPM [46]	w	81.35	81.11	81.22	81.28
BiMPM [46]+st.+LASER	w	81.10	82.31	81.70	81.15
BiMPM [46]+dy.+LASER	w	80.85	82.20	81.52	80.86
BiMPM [46]+st.+SB-BK	w	80.93	82.44	81.68	81.18
BiMPM [46]+dy.+SB-BK	w	81.09	81.73	81.41	80.92
BiMPM [46]+st.+SEMTS	w	82.16	81.30	81.73	81.77
BiMPM [46]+dy.+SEMTS	w	80.59	81.45	81.02	81.13

Table 7. The comparison between our method and other models on LCQMC.

Model	Emb	P	R	F	Acc.
CBOW [49]	c	66.5	82.8	73.8	70.6
CBOW [49]	w	67.9	89.9	77.4	73.7
CBOW [49]+st.+LASER	w	67.75	70.64	77.65	75.05
CBOW [49]+dy.+LASER	w	68.21	70.64	77.51	74.93
CBOW [49]+st.+SB-BK	w	68.28	70.64	78.07	75.11
CBOW [49]+dy.+SB-BK	w	68.53	70.64	77.64	74.98
CBOW [49]+st.+SEMTS	w	68.24	70.64	78.13	75.37
CBOW [49]+dy.+SEMTS	w	67.65	70.64	77.92	75.16
Text-CNN [44]	c	67.1	85.6	75.2	71.8
Text-CNN [44]	w	68.4	84.6	75.7	72.8
Text-CNN [44]+st.+LASER	w	69.45	84.04	76.05	73.85
Text-CNN [44]+dy.+LASER	w	67.76	86.31	75.92	73.56
Text-CNN [44]+st.+SB-BK	w	70.84	82.62	76.28	73.90
Text-CNN [44]+dy.+SB-BK	w	68.37	85.93	76.15	73.51
Text-CNN [44]+st.+SEMTS	w	69.93	85.68	77.01	73.98
Text-CNN [44]+dy.+SEMTS	w	69.61	85.87	76.89	73.54
BiLSTM [50]	c	67.4	91.0	77.5	73.50
BiLSTM [50]	w	70.6	89.3	78.9	76.10
BiLSTM [48]+st.+LASER	w	70.97	89.49	79.16	77.14
BiLSTM [48]+dy.+LASER	w	71.16	88.83	79.02	77.02
BiLSTM [48]+st.+SB-BK	w	70.93	89.37	79.09	76.95
BiLSTM [48]+dy.+SB-BK	w	71.25	88.26	78.85	76.87
BiLSTM [48]+st.+SEMTS	w	71.07	89.84	79.36	77.31
BiLSTM [48]+dy.+SEMTS	w	71.51	88.59	79.14	77.18
BiMPM [46]	c	77.60	93.90	85.00	83.40
BiMPM [46]	w	77.70	93.50	84.90	83.30
BiMPM [46]+st.+LASER	w	78.07	96.17	86.18	84.89
BiMPM [46]+dy.+LASER	w	77.98	96.76	86.36	85.11
BiMPM [46]+st.+SB-BK	w	78.02	96.62	86.33	85.05
BiMPM [46]+dy.+SB-BK	w	78.87	94.18	85.85	84.88
BiMPM [46]+st.+SEMTS	w	79.20	95.53	86.60	**85.17**
BiMPM [46]+dy.+SEMTS	w	78.95	94.12	85.87	84.85

Figures 11 and 12 show the experimental results on BQ and LCQMC at different low-frequency word embedding replacement rates. The bars "SEMTS-BERT *" represent the accuracy of BiMPM at different low-frequency word embedding replacement rates when SEMTS-BERT is used to estimate the embeddings. The bars "LASER *" and "SBERT-WK *" also have similar meanings. "Static" and "dynamic" indicate that word embeddings are fixed and to be tuned during the training. When the low-frequency word embedding

replacement rate is 0, we did not use the word embeddings calculated by the sentence representation model to replace the original word embeddings estimated by Word2Vec.

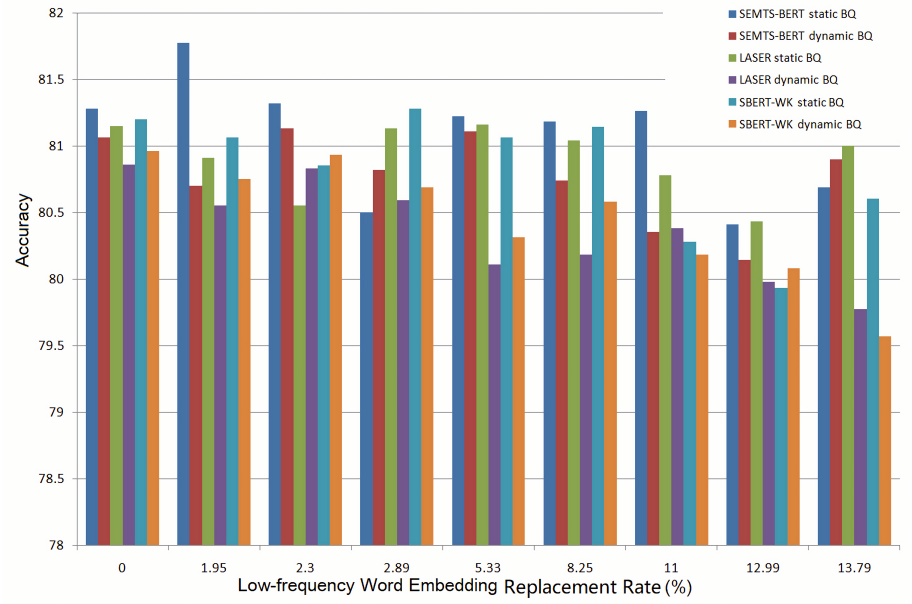

Figure 11. The influence of different low-frequency word embedding replacement rates on the performance of BiMPM evaluated on BQ.

It can be seen from Figure 11 that when using LASER and SBERT-WK to estimate OOVs and low-frequency words' embeddings, the replacement cannot improve the accuracy of BiMPM on BQ, no matter if the embeddings of words are static or dynamic. When using SEMTS-BERT to estimate OOVs and low-frequency words' embeddings and the embeddings of words are fixed during the evaluation, the replacement of low-frequency words' embeddings can improve the accuracy of the word-based BiMPM at the replacement rate of 1.95% (from 81.28% to 81.77%). When word embeddings are dynamic, SEMTS-BERT can not improve the performance of the word-based BiMPM on BQ either.

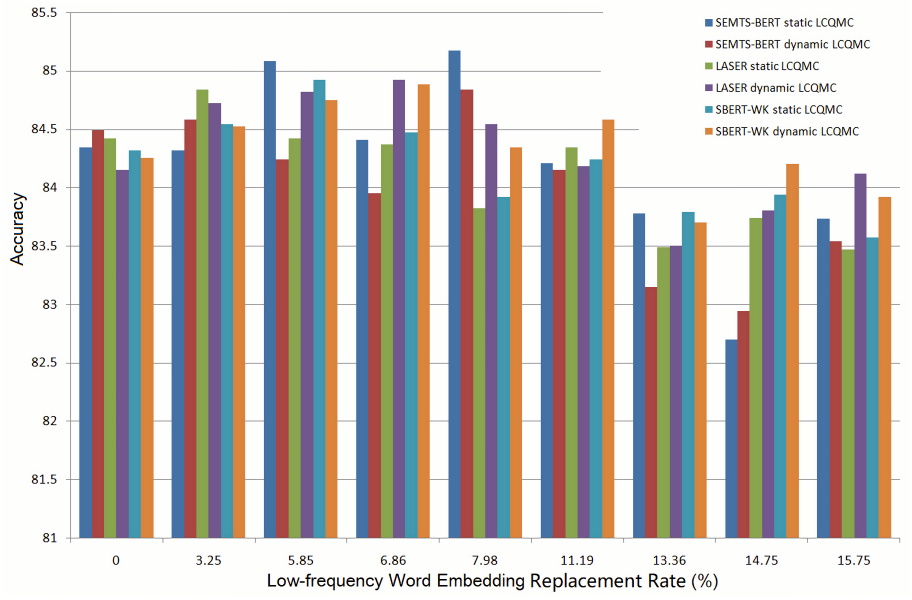

Figure 12. The influence of different low-frequency word embedding replacement rates on the performance of BiMPM evaluated on LCQMC.

The situation in Figure 12 is slightly different. On LCQMC, whether the word embeddings are dynamic or static, replacing low-frequency words' embeddings can improve the performance of the word-based BiMPM. However, as the replacement rate increases, the performance of word-based BiMPM decreases significantly. When using LASER to estimate low-frequency words' embeddings, the replacement can improve the accuracy of BiMPM at the rates of 3.25% (word embeddings are fixed) and 6.86% (word embedding are dynamic). We can draw the similar conclusion from the bars of SBERT-WK. When using SEMTS-BERT to estimate low-frequency words' embeddings and the word embeddings are static, the replacement can effectively improve the accuracy of BiMPM from 84.34% to 85.08% and from 84.34% to 85.17% at the replacement rate of 5.85% and 7.98%. The best accuracy of BiMPM on LCQMC in [38] is 83.34%, obtained by the character-based model. Our result is much better than the benchmark. When the word embeddings are dynamic, we can also draw a similar conclusion.

In addition, we obtain an interesting conclusion, which is shown in Figure 13. When low-frequency words' embeddings are not replaced, if the model has achieved good performance through the careful fine-tuning of hyper-parameters (the solid broken lines), then replacing low-frequency words' embeddings can no longer improve the performance. On the contrary, if the performance of the model is relatively poor when no low-frequency words' embeddings are replaced (the dashed broken lines), the replacement can improve the performance at some replacement rates. This conclusion can not only guide us to adjust hyper-parameters, but also enable us to obtain better performance by replacing the low-frequency words' embeddings. This is because either we have obtained good performance or we will obtain better performance by replacing the embeddings of low-frequency words.

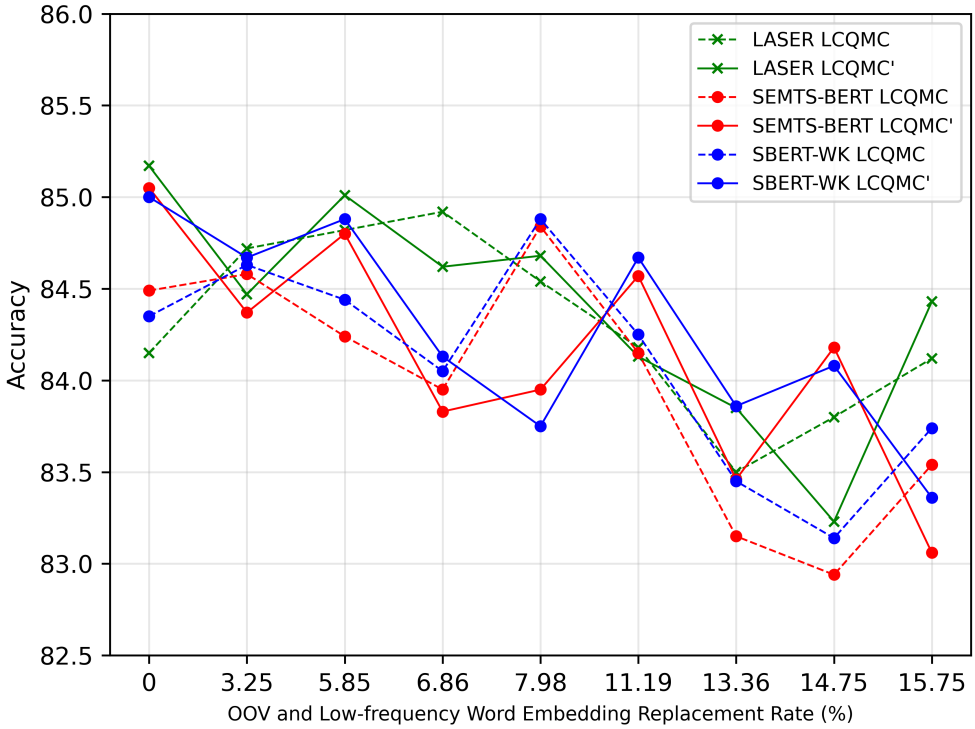

Figure 13. The influence of initial performances (low-frequency word embedding replacement rate is 0) on BiMPM performance at different replacement rates. We take evaluation on LCQMC as an example.

5.7.5. Analysis and Conclusion

Except for the condition that the word embeddings are static, replacing low-frequency words' embeddings can hardly improve the performance on the smaller BQ dataset (take the benchmark as a reference). This may be caused by sparsity. On the larger LCQMC, the sparsity has been greatly alleviated. Regardless of whether the word embeddings are

dynamic or static in the evaluation, replacing low-frequency words' embeddings estimated by all sentence representation models can improve the performances of word-based BiMPM, and we achieve new benchmark on LCQMC.

However, only a suitable low-frequency word embedding replacement rate can improve the performance. The performance will be reduced when the replacement rate is too high. From our experimental results, the smaller the dataset, the more limited the performance improvement obtained by the replacement of low-frequency words' embeddings. We think this is not only caused by sparsity, but also by the lack of coupling between the two different semantics spaces (the way that sentence representation models calculate word embeddings is different from the way that Word2Vec calculates word embeddings). In addition, if the model has achieved good performance when no low-frequency words' embeddings are replaced, the replacement cannot improve the performance. On the contrary, if the performance is not good when no low-frequency words' embeddings are replaced, the replacement will improve the performance.

In summary, we can draw two conclusions from the extrinsic evaluation. The first is that we can obtain higher-quality low-frequency word embeddings through our proposed method. When the dataset is large, replacing the original embeddings of low-frequency words in an appropriate proportion can improve the performance. The second is that SEMTS-BERT can represent the semantics of sentences well. This is because, on the BQ dataset, only SEMTS-BERT improves the performance of the word-based BiMPM, and on LCQMC, we achieve a new benchmark by using the embeddings estimated by SEMTS-BERT to replace the original embeddings of low-frequency words.

6. Discussion

The sparsity makes the semantics of words and phrases not fully learned, which in turn harms the performance of NLP tasks [16,17,22,25]. To reduce the sparsity, researchers have designed effective algorithms to split long words into short fragments. These algorithms include BPE and WordPiece [1,2,20]. There is also a study pointing out that using character-level models in Chinese NLP tasks can achieve better performance [16]. In this article, we use the dictionary to estimate the embeddings of low-frequency words. In extrinsic tasks, we obtain better performance by using word embeddings estimated by our proposed method to replace the original low-frequency words' embeddings (estimated by Word2Vec). This shows that our method can provide higher-quality low-frequency word embedding. However, from the experimental results, we can see that too-high a replacement rate will harm the performance of tasks. In addition, performing such a replacement on a larger dataset will lead to higher performance improvement.

In this article, we design a new sentence representation model and expect to extract the semantics of explanatory notes more efficiently. Our sentence representation model achieves the best performance in many tasks in both the intrinsic and extrinsic experiments.

In summary, dealing with OOVs and low-frequency words is one of the challenges in NLP tasks. OOVs and low-frequency words are universal. Therefore, we think that it is very difficult to eliminate the OOV problem. Although the method proposed in this paper reduces the impact of the OOV problem on performance to a certain extent, there are still many problems worthy of further study. In the future, we will conduct in-depth research in the following aspects:

- Use relationships between the rich nodes in knowledge bases to estimate the embedding of low-frequency words.
- Construct more high-performance sentence representation models to extract semantics from sentences.
- Since there are two different word embedding estimation methods, we will study measures to make two semantics spaces better coupled.
- Use the correspondence between words in multilingual dictionaries to estimate the embeddings of low-resource language words.

Author Contributions: Conceptualization, Y.H.; methodology, X.L.; software, X.L.; validation, C.W., C.Z. and Y.D.; formal analysis, Y.H.; writing, K.Y.; funding acquisition, Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China under Grant No. 61866008, and we are also supported by the Basic and Applied Basic Research Fund of Guangdong Province, China under Grant No. 2019B1515120085. We are also supported by the central government guides for the local science and technology development fund project subsidization (Guike AD20238072).

Acknowledgments: We are grateful to the anonymous reviewers for their insightful suggestions and the editors of Applied Sciences—Basel.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Evaluation Results of Other Methods on BQ

Table A1. The evaluation results of other methods on BQ. 1L-MLP stands for a one-layer MLP, LR stands for logistic regression [51], and DCNN stands for a dynamic convolutional neural network with k-max pooling [47]. When SVM, MLP, and LR are used for evaluation, the sentence representation is obtained by implementing average pooling transformation on the variable-length word (character) embedding sequences.

Model	Emb	P	R	F	Acc.
RNN	c	66.23	64.14	65.17	64.53
RNN	w	65.78	61.86	63.76	63.09
RNN+st.+LASER	w	65.53	64.38	64.95	64.49
RNN+dy.+LASER	w	63.81	65.45	64.62	63.86
RNN+st.+SB-BK	w	65.22	64.33	64.77	64.25
RNN+dy.+SB-BK	w	66.36	63.73	65.02	64.01
RNN+st.+SEMTS	w	65.69	64.58	65.13	64.61
RNN+dy.+SEMTS	w	64.75	64.89	64.82	64.17
LSTM [42]	c	68.62	70.57	69.58	70.56
LSTM [42]	w	67.15	67.61	67.38	68.02
LSTM [42]+st.+LASER	w	70.61	71.52	71.06	70.10
LSTM [42]+dy.+LASER	w	68.76	70.93	69.83	69.28
LSTM [42]+st.+SB-BK	w	70.23	70.33	70.28	70.27
LSTM [42]+dy.+SB-BK	w	69.54	71.73	70.62	69.34
LSTM [42]+st.+SEMTS	w	68.56	73.38	70.89	71.33
LSTM [42]+dy.+SEMTS	w	68.27	71.27	69.74	70.16
SVM [52]	c	62.87	63.43	63.15	63.39
SVM [52]	w	63.98	63.90	63.94	63.71
SVM [52]+st.+LASER	w	64.52	63.08	63.79	64.56
SVM [52]+dy.+LASER	w	65.85	62.44	64.10	64.17
SVM [52]+st.+SB-BK	w	64.76	63.77	64.26	63.73
SVM [52]+dy.+SB-BK	w	63.94	63.78	63.86	63.45
SVM [52]+st.+SEMTS	w	64.92	62.31	63.59	64.11
SVM [52]+dy.+SEMTS	w	65.27	63.42	64.33	63.78
1L-MLP	c	65.83	62.71	64.23	63.89
1L-MLP	w	64.96	62.64	63.78	63.26
1L-MLP+st.+LASER	w	65.34	63.60	64.46	63.73
1L-MLP+dy.+LASER	w	64.91	63.86	64.38	63.69
1L-MLP+st.+SB-BK	w	64.37	63.61	63.99	63.88
1L-MLP+dy.+SB-BK	w	63.95	64.47	64.21	63.57
1L-MLP+st.+SEMTS	w	64.54	65.04	64.79	63.95
1L-MLP+dy.+SEMTS	w	64.89	62.59	63.72	63.82

Table A1. *Cont.*

Model	Emb	P	R	F	Acc.
LR [51]	c	63.81	62.00	62.89	63.65
LR [51]	w	63.79	61.40	62.57	62.81
LR [51]+st.+LASER	w	63.55	62.20	62.87	63.52
LR [51]+dy.+LASER	w	64.27	62.17	63.20	63.76
LR [51]+st.+SB-BK	w	65.16	62.15	63.62	63.85
LR [51]+dy.+SB-BK	w	64.56	61.30	62.89	63.49
LR [51]+st.+SEMTS	w	63.93	63.59	63.76	63.83
LR [51]+dy.+SEMTS	w	64.25	61.66	62.93	63.58
DCNN [47]	c	72.68	70.09	71.36	70.12
DCNN [47]	w	70.52	68.55	69.52	68.47
DCNN [47]+st.+LASER	w	71.24	70.01	70.62	70.01
DCNN [47]+dy.+LASER	w	70.95	70.63	70.79	69.37
DCNN [47]+st.+SB-BK	w	70.29	69.44	69.86	69.79
DCNN [47]+dy.+SB-BK	w	71.57	69.52	70.53	69.25
DCNN [47]+st.+SEMTS	w	72.63	69.59	71.08	70.26
DCNN [47]+dy.+SEMTS	w	71.28	69.25	70.25	69.88

Appendix B. Evaluation Results of Other Methods on LCQMC

Table A2. The evaluation results of other methods on BQ. 1L-MLP stands for a one-layer MLP, LR stands for logistic regression [51], and DCNN stands for a dynamic convolutional neural network with k-max pooling [47]. When SVM, MLP, and LR are used for evaluation, the sentence representation is obtained by implementing average pooling transformation on the variable-length word (character) embedding sequences.

Model	Emb	P	R	F	Acc.
RNN	c	67.68	66.04	66.85	65.36
RNN	w	71.53	67.36	69.38	67.27
RNN+st.+LASER	w	72.86	69.58	71.18	69.23
RNN+dy.+LASER	w	71.95	69.72	70.82	69.09
RNN+st.+SB-BK	w	71.52	69.63	70.56	68.77
RNN+dy.+SB-BK	w	73.17	69.05	71.05	68.85
RNN+st.+SEMTS	w	72.76	69.53	71.11	69.36
RNN+dy.+SEMTS	w	71.25	70.22	70.73	69.24
LSTM [42]	c	74.81	71.79	73.27	70.25
LSTM [42]	w	75.96	73.44	74.68	73.51
LSTM [42]+st.+LASER	w	77.08	74.37	75.70	74.53
LSTM [42]+dy.+LASER	w	77.35	75.18	76.25	74.11
LSTM [42]+st.+SB-BK	w	78.12	74.33	76.18	74.69
LSTM [42]+dy.+SB-BK	w	77.73	73.56	75.59	73.87
LSTM [42]+st.+SEMTS	w	78.17	75.34	76.73	74.82
LSTM [42]+dy.+SEMTS	w	78.09	74.70	76.36	74.66
SVM [52]	c	66.39	65.20	65.79	64.35
SVM [52]	w	68.95	67.11	68.02	66.77
SVM [52]+st.+LASER	w	69.86	68.57	69.21	67.64
SVM [52]+dy.+LASER	w	71.57	69.50	70.52	67.92
SVM [52]+st.+SB-BK	w	72.63	69.86	71.22	68.16
SVM [52]+dy.+SB-BK	w	71.61	69.77	70.68	67.83
SVM [52]+st.+SEMTS	w	72.72	69.61	71.13	68.07
SVM [52]+dy.+SEMTS	w	71.68	69.15	70.39	67.95

Table A2. *Cont.*

Model	Emb	P	R	F	Acc.
1L-MLP	c	66.27	63.55	66.10	64.88
1L-MLP	w	67.53	64.35	68.06	65.90
1L-MLP+st.+LASER	w	67.86	66.47	68.85	67.16
1L-MLP+dy.+LASER	w	68.49	65.73	69.27	67.08
1L-MLP+st.+SB-BK	w	68.65	65.14	68.26	66.85
1L-MLP+dy.+SB-BK	w	67.82	66.06	68.52	66.93
1L-MLP+st.+SEMTS	w	69.38	65.28	70.20	67.27
1L-MLP+dy.+SEMTS	w	68.94	65.53	69.53	67.19
LR [51]	c	65.97	65.43	65.70	64.23
LR [51]	w	68.92	65.62	67.23	65.78
LR [51]+st.+LASER	w	69.37	67.38	68.36	66.98
LR [51]+dy.+LASER	w	71.16	67.52	69.29	67.03
LR [51]+st.+SB-BK	w	70.95	66.44	68.62	66.85
LR [51]+dy.+SB-BK	w	71.29	69.22	70.24	66.96
LR [51]+st.+SEMTS	w	72.31	69.89	71.08	67.16
LR [51]+dy.+SEMTS	w	71.15	70.61	70.88	67.21
DCNN [47]	c	75.61	72.88	74.22	72.50
DCNN [47]	w	76.73	73.63	75.15	73.49
DCNN [47]+st.+LASER	w	78.69	75.90	77.27	75.31
DCNN [47]+dy.+LASER	w	79.35	76.66	78.15	75.23
DCNN [47]+st.+SB-BK	w	78.13	75.57	76.83	75.56
DCNN [47]+dy.+SB-BK	w	79.57	74.95	77.19	75.38
DCNN [47]+st.+SEMTS	w	79.38	77.28	78.31	75.45
DCNN [47]+dy.+SEMTS	w	78.64	77.10	77.86	75.29

References

1. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
2. Language Models Are Few-Shot Learners. Available online: https://arxiv.org/pdf/2005.14165.pdf (accessed on 1 February 2021).
3. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Lake Tahoe, CA, USA, 5–10 December 2013; pp. 3111–3119.
4. Liu, P.; Qiu, X.; Huang, X. Recurrent Neural Network for Text Classification with Multi-Task Learning. In Proceedings of the International Joint Conferences on Artifical Intelligence (IJCAI), New York, NY, USA, 9–15 July 2016; pp. 2873–2879.
5. Joulin, A.; Grave, E.; Bojanowski, P.; Mikolov, T. Bag of Tricks for Efficient Text Classification. In Proceedings of the European Chapter of the Association for Computational Linguistics (EACL), Valencia, Spain, 3–7 April 2017; pp. 427–431.
6. Yao, L.; Mao, C.; Luo, Y. Graph Convolutional Networks for Text Classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Hawaii, HI, USA, 27 January–1 February 2019; pp. 7370–7377.
7. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All You Need. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 1–11.
8. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, 7–12 August 2016; pp. 1715–1727.
9. Sutskever, I.; Vinyals, O.; Quoc, V.L. Sequence to Sequence Learning with Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Quebec City, QC, Canada, 8–11 December 2014; pp. 1–9.
10. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by backpropagating errors. *Nature* **1986**, *6088*, 533–536. [CrossRef]
11. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
12. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. In Proceedings of the 1st International Conference on Learning Representations (ICLR), Scottsdale, AZ, USA, 2–4 May 2013; pp. 1–12.
13. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
14. Petersy, M.E.; Neumanny, M.; Iyyery, M.; Gardnery, M.; Clark, C.; Lee, K.; Zettlemoyery, L. Deep contextualized word representations. In Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL), New Orleans, LA, USA, 1–6 June 2018; pp. 1532–1543.

15. Huang, K.; Huang, D.; Liu, Z.; Mo, F. A Joint Multiple Criteria Model in Transfer Learning for Cross-domain Chinese Word Segmentation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Virtual Conference, 16–20 November 2020; pp. 16–20.
16. Meng, Y.; Li, X.; Sun, X.; Han, Q.; Yuan, A.; Li, J. Is Word Segmentation Necessary for Deep Learning of Chinese Representation? In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 28 July–2 August 2019; pp. 3242–3252.
17. Hu, Z.; Chen, T.; Chang, K.; Sun, Y. Few-Shot Representation Learning for Out-Of-Vocabulary Words. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 28 July–2 August 2019; pp. 4102–4112.
18. Liu, J.; Wu, F.; Wu, C.; Huang, Y.; Xie, X. Neural chinese word segmentation with dictionary. *Neurocomputing* **2019**, *338*, 46–54. [CrossRef]
19. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Vancouver, BC, Canada, 8–12 December 2019; pp. 5753–5763.
20. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. Available online: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf (accessed on 25 July 2021).
21. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
22. Patel, R.; Domeniconi, C. Estimator Vectors: OOV Word Embeddings based on Subword and Context Clue Estimates. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
23. Ha, P.; Zhang, S.; Djuric, N.; Vucetic, S. Improving Word Embeddings through Iterative Refinement of Word- and Character-level Models. In Proceedings of the 28th International Conference on Computational Linguistics, Online Conference, 8–13 December 2020; pp. 1204–1213.
24. Fukuda, N.; Yoshinaga, N.; Kitsuregawa, M. Robust Backed-off Estimation of Out-of-Vocabulary Embeddings. In Proceedings of the Association for Computational Linguistics: EMNLP 2020, Virtual Conference, 16–20 November 2020; pp. 4827–4838.
25. Garneau, N.; Leboeuf, J.; Lamontagne, L. Contextual Generation of Word Embeddings for out of Vocabulary Words in Downstream Tasks. In *Advances in Artificial Intelligence*; Meurs, M.J., Rudzicz, F., Eds.; Springer: Berlin, Germany, 2019; pp. 563–569.
26. Pagliardini, M.; Gupta, P.; Jaggi, M. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics-Human Language Technologies (NAACL-HLT), New Orleans, LA, USA, 1–6 June 2018; pp. 528–540.
27. Wang, B.; Kuo, C.-C.J. SBERT-WK: A Sentence Embedding Method by Dissecting BERT-Based Word Models. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2020**, *28*, 2146–2157. [CrossRef]
28. Schwenk, H.; Douze, M. Learning Joint Multilingual Sentence Representations with Neural Machine Translation. In Proceedings of the 2nd Workshop on Representation Learning for NLP, Vancouver, BC, Canada, 4 August 2017; pp. 157–167.
29. Nie, A.; Bennett, E.D.; Goodman, N.D. DisSent: Learning Sentence Representations from Explicit Discourse Relations. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 28 July–2 August 2019; pp. 4497–4510.
30. Cui, Y.; Che, W.; Zhang, W.; Liu, T.; Wang, S.; Hu, G. Discriminative Sentence Modeling for Story Ending Prediction. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 7602–7609.
31. Liu, B.; Wang, L.; Yin, G. Learning distributed sentence vectors with bi-directional 3D convolutions. In Proceedings of the 28th International Conference on Computational Linguistics, Online Conference, 8–13 December 2020; pp. 6820–6830.
32. LASER: Language-Agnostic SEentence Representation. Available online: https://github.com/facebookresearch/LASER (accessed on 25 January 2021).
33. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. Available online: https://arxiv.org/pdf/1907.11692.pdf (accessed on 10 May 2021).
34. Yang, Z.; Yan, D.; Dyer, C.; He, X.; Smola, A.; Hovy, E. Hierarchical Attention Networks for Document Classification. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, 7–12 August 2016; pp. 1480–1489.
35. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015; pp. 1–12.
36. Krasnowska-Kieras, K.; Wróblewska, A. Empirical Linguistic Study of Sentence Embeddings. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 28 July–2 August 2019; pp. 5729–5739.
37. Chen, J.; Chen, Q.; Liu, X.; Yang, H.; Lu, D.; Tang, B. The BQ Corpus: A Large-scale Domain-specific Chinese Corpus For Sentence Semantic Equivalence Identification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, 31 October–4 November 2018; pp. 4946–4951.
38. Liu, X.; Chen, Q.; Deng, C.; Zeng, H.; Chen, J.; Li, D.; Tang, B. LCQMC: A Large-scale Chinese Question Matching Corpus. In Proceedings of the International Conference on Computational Linguistics (COLING), Santa Fe, NM, USA, 20–26 August 2018; pp. 1952–1962.

39. Conneau, A.; Kruszewski, G.; Lample, G.; Barrault, L.; Baroni, M. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Melbourne, Australia, 15–20 July 2018; pp. 2126–2136.
40. Conneau, A.; Kiela, D. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018), Miyazaki, Japan, 7–12 May 2018; pp. 1699–1704.
41. Conneau, A.; Rinott, R.; Lample, G.; Williams, A.; Bowman, S.; Schwenk, H.; Stoyanov, V. XNLI: Evaluating Cross-lingual Sentence Representations. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, 31 October–4 November 2018; pp. 2475–2485.
42. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
43. Maaten, L.V.; Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
44. Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
45. Gong, Y.; Luo, H.; Zhang, J. Natural language inference over interaction space. In Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018; pp. 1–10.
46. Wang, Z.; Hamza, W.; Florian, R. Bilateral Multi-Perspective Matching for Natural Language Sentences. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Melbourne, Australia, 19–25 August 2017; pp. 4144–4150.
47. Kalchbrenner, N.; Grefenstette, E.; Blunsom, P. A Convolutional Neural Network for Modelling Sentences. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Baltimore, MD, USA, 22–27 June 2014; pp. 655–665.
48. Gravesa, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef] [PubMed]
49. Yin, W.; Schütze, H. Discriminative Phrase Embedding for Paraphrase Identification. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, CO, USA, 31 May–5 June 2015; pp. 1368–1373.
50. Tomar, G.S.; Duque, T.; Täckström, O.; Uszkoreit, J.; Das, D. Neural Paraphrase Identification of Questions with Noisy Pretraining. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Copenhagen, Denmark, 9–11 September 2017; pp. 142–147.
51. Le, Q.; Mikolov, T. Distributed Represenations of Sentences and Documents. In Proceedings of the International Conference on Machine Learning (ICML 2014), Beijing, China, 21–26 June 2014; pp. 1188–1196.
52. Silva, J.; Coheur, L.; Mendes, A.C.; Wichert, A. Wichert. From symbolic to sub-symbolic information in question classification. *Artif. Intell. Rev.* **2011**, *35*, 137–154. [CrossRef]