applied sciences

MDPI

Article

A New Sentiment-Enhanced Word Embedding Method for Sentiment Analysis

Qizhi Li, Xianyong Li *, Yajun Du, Yongquan Fan and Xiaoliang Chen

School of Computer and Software Engineering, Xihua University, Chengdu 610039, China
* Correspondence: lixy@mail.xhu.edu.cn or xian-yong@163.com

Abstract: Since some sentiment words have similar syntactic and semantic features in the corpus, existing pre-trained word embeddings always perform poorly in sentiment analysis tasks. This paper proposes a new sentiment-enhanced word embedding (S-EWE) method to improve the effectiveness of sentence-level sentiment classification. This sentiment enhancement method takes full advantage of the mapping relationship between word embeddings and their corresponding sentiment orientations. This method first converts words to word embeddings and assigns sentiment mapping vectors to all word embeddings. Then, word embeddings and their corresponding sentiment mapping vectors are fused to S-EWEs. After reducing the dimensions of S-EWEs through a fully connected layer, the predicted sentiment orientations are obtained. The S-EWE method adopts the cross-entropy function to calculate the loss between predicted and true sentiment orientations, and backpropagates the loss to train the sentiment mapping vectors. Experiments show that the accuracy and macro-F1 values of six sentiment classification models using Word2Vec and GloVe with the S-EWEs are on average 1.07% and 1.58% higher than those without the S-EWEs on the SemEval-2013 dataset, and on average 1.23% and 1.26% higher than those without the S-EWEs on the SST-2 dataset. In all baseline models with S-EWEs, the convergence time of the attention-based bidirectional CNN-RNN deep model (ABCDM) with S-EWEs was significantly decreased by 51.21% of ABCDM on the SemEval-2013 dataset. The convergence time of CNN-LSTM with S-EWEs was vastly reduced by 41.34% of CNN-LSTM on the SST-2 dataset. In addition, the S-EWE method is not valid for contextualized word embedding models. The main reasons are that the S-EWE method only enhances the embedding layer of the models and has no effect on the models themselves.

Keywords: sentiment enhancement; sentiment-enhanced word embedding; sentiment lexicon; word embedding

Citation: Li, Q.; Li, X.; Du, Y.; Fan, Y.; Chen, X. A New Sentiment-Enhanced Word Embedding Method for Sentiment Analysis. *Appl. Sci.* **2022**, *12*, 10236. https://doi.org/10.3390/app122010236

Academic Editor: Emilio Soria-Olivas

Received: 18 August 2022
Accepted: 8 October 2022
Published: 11 October 2022

1. Introduction

Word embedding maps a word into a vector space, playing an important role in natural language processing (NLP). Word vectors contain rich information, such as local contextual information [1], global co-occurrence information [2], and global contextual information [3]. Moreover, word embedding can apply to many NLP tasks, such as sentiment analysis [4–8], part-of-speech tagging [9], and named entity recognition [10]. It can also apply to many interdisciplinary tasks, such as influence maximization [11] and emotion role identification [12].

Word embedding learned by deep learning carries contextual semantic and syntactic information, which is beneficial for NLP tasks. Sitaula et al. [13] evaluated many machine learning and deep learning methods on sentiment analysis tasks. However, for feature-based word embedding, such as Word2Vec [1], if two words with opposite sentiment polarity have similar context in a corpus, the performance of word embedding may not be so good [14,15]. This is because feature-based word embeddings only assign a unique word embedding to each word and cannot generate word embeddings based on the context in

downstream tasks. For example, the words "good" and "terrible" from two different sentences, i.e., "The weather is good today" and "The weather is terrible today," have similar contexts in a corpus and have similar word embeddings. In this situation, sentiment classification models cannot determine which is a positive or negative sentence. Yu et al. [15] concluded that about 30% of the top 10 semantically similar words have opposite sentiment orientations. In most tasks, sentiments could divide into two (positive, negative) or three dimensions (positive, neutral, and negative). Ekman [16] even divided sentiments into six categories (anger, disgust, fear, happiness, sadness, and surprise). Based on the six categories, Xu [17] further divided sentiments into seven categories (anger, disgust, fear, sadness, happiness, good, and surprise). The latest research of Google [18] found that sentiments could divide into 28 categories. However, the sentiment dimensions are also much fewer than the dimensions of word embedding. The dimensions of Word2Vec [1] and global vectors for word representation (GloVe) [2] are usually 300; the dimensions of the bidirectional encoder representations from the transformers base (BERT-base) [3] number 768; and the dimensions of BERT-large [3] number 1024. Thus, the dimensions of sentiment and word embedding are different.

Sentiment enhancement is a way of injecting sentiments into word embedding, making word embedding contain sentiment information. However, sentiment enhancement concerning sentiment and word embedding faces two challenges (see Figure 1).

1. Due to the different dimensions of sentiment word embeddings and sentiment orientations, they are hard to fuse into one vector.
2. Since sentiment and word embeddings belong to two different vector spaces, sentiment classification models could not directly operate on them.

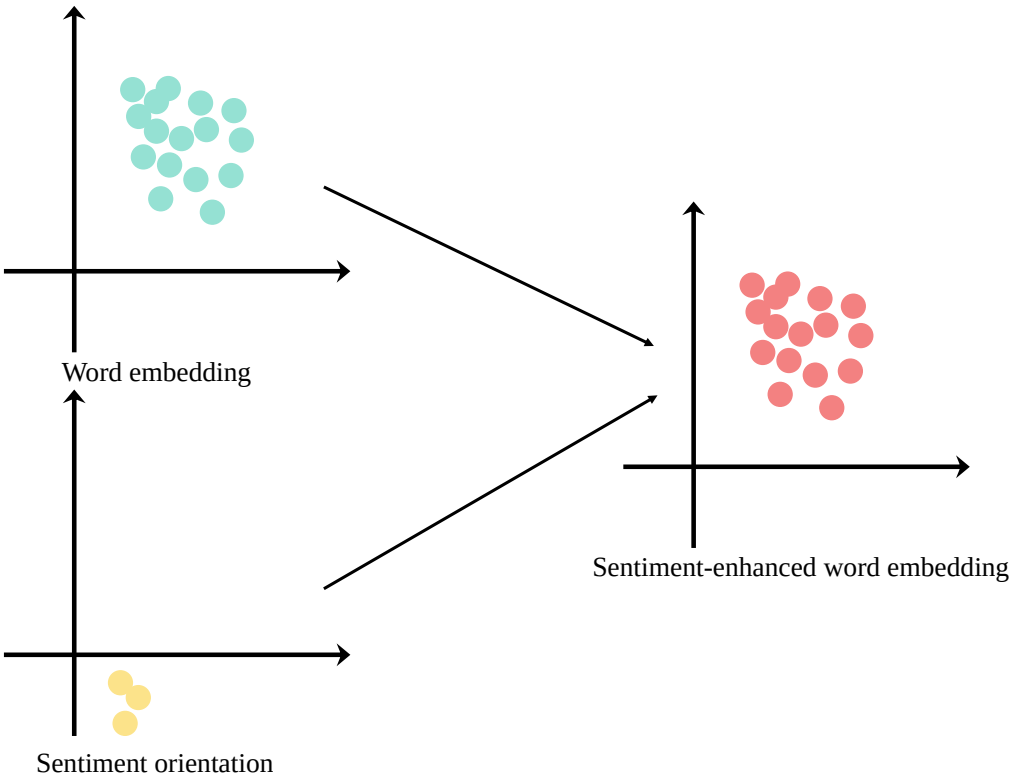


Figure 1. Intuitiveillustration of a sentiment-enhanced word embedding. Vectors with different colors represent that they belong to different vector spaces.

To solve the above problems, we expect to find a method to build the mapping relationship between words and their sentiment orientations in sentiment lexicons and fuse sentiment information into these words. Fortunately, we are inspired by translations in

the embedding space (TransE) [19] and contrastive learning [20,21]. TransE is a knowledge graph embedding method. It shortens the distance between two related entities and increases the distance between two unrelated entities in vector space. Meanwhile, contrastive learning narrows the distance between positive examples and increases the distance between negative examples in vector space. The bootstrap your own latent (BYOL) [20] minimizes the distance between two similar images; the supervised contrastive pre-training (SCAPT) [21] used supervised contrastive learning to cluster explicit and implicit positive sentiments, cluster explicit and implicit negative sentiments, and separated these two clusters. By borrowing these ideas, we propose a novel sentiment-enhanced word embedding (S-EWE) method to improve the performances of sentiment classification models. Specifically, we convert words in the sentiment lexicon to word embeddings and arrange each word embedding for a sentiment mapping vector. Then, we add original word embeddings and their sentiment mapping vectors to obtain the sentiment-enhanced word embeddings with the information of word and sentiment. Thirdly, we adopt one fully connected layer to reduce the dimensions of sentiment-enhanced word embedding to get the predicted sentiment orientations. Finally, we calculate the loss of the predicted and true sentiment orientations by the cross-entropy function. We further train the sentiment mapping vectors through backpropagating the loss so that the trained sentiment mapping vectors can find the mapping relationship between words in sentiment lexicons and their sentiment orientations. Under different sentiment classification models, we confirmed the effectiveness of the S-EWE method. The main contributions are as follows:

1. A new sentiment-enhanced word embedding method is proposed. This method establishes the mapping relationship between words and their sentiment orientations by vector addition.
2. Abundant experiments were performed on TextCNN [22], TextRNN [23], TextRCNN [24], TextBiRCNN [24], an attention-based bidirectional CNN-RNN deep model (ABCDM) [25], and CNN-LSTM [26] using Word2Vector [1]; and GloVe [2] with or without sentiment-enhanced word embeddings on the semantic evaluation 2013 (SemEval-2013) dataset [27] and the Stanford sentiment treebank (SST-2) dataset [28]. Experimental results show that sentiment-enhanced word embeddings can improve the ability of these models' sentiment classification.

The subsequent parts of this paper are organized as follows. Section 2 introduces the related work. The sentiment-enhanced word embedding method is shown in Section 3. Datasets, empirical results, and model analysis are presented in Section 4. Finally, we summarize our paper and put forward a future outlook in Section 5.

2. Related Work

2.1. Word Embedding

Word embedding is a knowledge enhancement technique because it can capture syntactic and semantic information in textual corpora. Word2Vec [1] has two models: a continuous skip-gram model (Skip-gram) and a continuous bag-of-words model (CBOW). They all use sliding windows to capture contextual information between words. Word2Vec achieved the best results on the Semantic-Syntactic Word Relationship dataset and the Microsoft Sentence Completion Challenge dataset. However, Word2Vec could not get global information of words. GloVe [2] used a co-occurrence probability matrix to obtain the global information of words. It also used slide windows to capture contextual information between words. Experimental results showed that GloVe achieved state-of-the-art performance on word analogies, similarity, and named entity recognition tasks. Since word embeddings trained by Word2Vec and GloVe could only assign a vector to each word, they could not solve the ambiguity problem of words. In addition, Word2Vec and GloVe could only capture the contextual information of words in a window due to the limited computing power. With the increased computing power, the embedding from language models (ELMo) [29] used two reversed recurrent neural network (RNN) layers to encode one input sequence. For a word i, the forward RNN captured the above information of the word i, and the

backward RNN captured the following information of the word i. By concatenating the forward hidden state and backward hidden state of the word i, the contextual information of the word i was obtained. The ELMo model achieved remarkable performance on CoNLL 2003 NER and CoNLL 2000 chunking tasks.

Due to the improvements in computing power, BERT [3] used bidirectional transformers [30] to encode information of a whole sentence. Since the self-attention mechanism [30] can capture the information of an entire sequence, the contextual information between words is no longer limited to the sliding window. BERT broke all records for the general language understanding evaluation (GLUE), with an average improvement of 6.7% over the other models. Moreover, BERT achieved significant performance on SQuAD, CoNLL-2003, and SWAG datasets. Liu et al. [31] averaged Chinese and English word embeddings in multilingual BERT (m-BERT) [3], respectively. They obtained the Chinese–English difference vector by subtracting the average English vector from the average Chinese vector. The experimental results showed that a Chinese word embedding minus the difference vector could get English word embedding and vice versa.

Even though word embedding has shown its power in many natural language processing tasks, it cannot distinguish the sentiment orientation of semantically similar words well. Singh et al. [32] extracted hashtags, mentions, and keywords from tweets. They performed centrality-aware random walks on them to get their word/node embeddings. Then, they put the embeddings into a deep learning model for sentiment classification. The performance of their method is better than those of the original baseline models. Sent2Vec [33] used the smoothed inverse frequency (SIF) [34] or the unsupervised smoothed inverse frequency (uSIF) [35] to encode a sentence into a vector. Then, this vector is input into a multi-layer perceptron (MLP) to learn sentimental embedding. These methods could achieve good results based on the syntactic and semantic knowledge inside a dataset in tasks of different sentiment classifications. However, these methods could not utilize external knowledge about the dataset.

2.2. Knowledge Enhancement

Knowledge enhancement is a way of injecting external knowledge into models, improving the performances of the models on downstream tasks. Liang and Yi [36] proposed the two-stage three-way enhanced technique for inclusive policy text classification. They first used the ensemble convolution neural network to extract text representation in the first stage. Then, they used a new determination method to reduce the decision risk. If the classification accuracy at the first state was not within the confidence interval, they used a traditional machine learning technique to reclassify texts. They conducted some experiments on Chengdu and Xiamen datasets. Their experimental results showed that their method outperforms baseline models. Bai et al. [37] proposed the multi-view document clustering with enhanced semantic embedding (MDCE) to solve the problems, including the sparseness, the high dimensionality, and the inconsistency of document clustering. This model used two deep neural networks to obtain document representation and neighbors representation. Then, the model adopted one fusion layer to fuse different representations and used a clustering layer to output the cluster results. The proposed MDCE model achieved the best performance on Aminer, Aminer (700), 3-sources, Reuters, and Multi-source news datasets. ERNIE [38] incorporated knowledge graphs (KGs) into BERT [3]. It used two encoders [30] to encode tokens and KGs obtained by TransE [19], respectively. It also adopted a fusion layer to fuse these two representations into the same vector space. This method achieved excellent performance on FIGER, Open Entity, FewRel, and TACRED datasets. It also obtained comparable results with BERT in GLUE. However, it took much time in the process of entity alignment. To shorten the running time of entity alignment, the knowledge embedding and pre-trained language representation (KEPLER) [39] used the robustly optimized BERT approach (RoBERTa) [40] to represent definitions of considerable knowledge in knowledge graphs. This method combined knowledge graph and text representations, achieving better performance than ERNIE on TACRED, FewRel, and OpenEntity

datasets. Saxena et al. [41] proposed a temporal question answering system. They first extracted knowledge from temporal knowledge graphs. Then, they used the proposed temporal KGE model to obtain temporal knowledge representation. They further used BERT [3] to get text representation. Finally, they fused temporal knowledge representation and text representation to answer questions. Their model achieved dramatic performance on the CRONQUESTIONS dataset. Even though injecting external knowledge into models could improve the performances of different models [38,42,43], all the above models have not adopted sentiment information.

2.3. Sentiment Enhancement

In recent times, some scholars have used the sentiments as external knowledge. Shi et al. [44] proposed a sentiment-enhanced neural graph recommender. This model applied an attention mechanism [30] to extract a representation of a user's review and an item review representation. They further extracted another user representation and item representation based on a graph convolutional network (GCN) [45]. The model obtained the final user and item representations by concatenating the two user representations and item representations. Then, the model adopted a fully connected network to extract an auxiliary sentiment representation from the output of the self-attention network. Finally, they trained their sentiment-enhanced neural graph recommender with a loss between sentiment auxiliary representation and user-item representation. Their experiments achieved the best performance on Toys, Kindle, Yelp2017, and Yelp2018 datasets. However, this model only adopted sentiment as an additional feature to implement a recommendation system. It did not enhance word embedding with sentiments. Gavilanes et al. [46,47] proposed an unsupervised system with sentiment propagation across dependencies (USSPAD) model [48]. Their method found the relationship between each emoji's description and sentiment orientation. The classification results of their model are close to those of manual annotations. Yu et al. [15] used k-nearest neighbors (KNN) to zoom in on positive samples and pushed out negative samples in a sentiment lexicon. Their refinement method outperformed original baseline models on SemEval-2013 and SST-2 datasets. Wei et al. [49] used sentiment lexicon as external sentiment knowledge and injected this knowledge into the bidirectional long short-term memory (BiLSTM) to analyze implicit sentiment. They achieved the best results on SMP2019-ECISA, COAE (2015), and SemEval (2013–2017) datasets. Wang et al. [6] injected sentiment lexicon and labeled sentiment corpora as the pre-training corpora for two bidirectional gate recurrent unit (BiGRU) layers. They pre-trained their model with three objectives: predicting target words, predicting word sentiment, and predicting sentence sentiment. After pre-training, they concatenated word embeddings and the hidden states of two BiGRUs, and obtained the sentiment-enhanced word embedding. The experimental results show that their model achieved comparable results to contextualized word embedding models (such as BERT) on three datasets with only 1–7 M parameters. However, these models considered sentiment an additional feature or combined other features to analyze sentiment, but fewer scholars injected sentiment information into word embeddings.

In the field of injecting sentiment into word embedding, Sent2Vec [33] used SIF [34] or uSIF [35] to encode a sentence into a vector. The vector is further input into a fully connected network to learn sentimental embedding. This method achieved better results than other word embeddings on Dbpedia and Yahoo datasets. Naderalvojoud and Sezer [14] proposed Approaches 1 and 2 for enhancing sentiment. Specifically, Approach 1 adopted an MLP to find the mapping relationships between words and their sentiment intensity; and Approach 2 trained Word2Vec [1] and GloVe [2] with an additional sentiment vector if a target word is a sentiment word. The two methods achieved better results than Word2Vec and GloVe on SemEval-2013 and SST datasets. Therefore, few studies are focused on injecting sentiment information into word vectors.

3. The Proposed Method and Its Applications on Downstream Tasks

This section illustrates a novel sentiment-enhanced word embedding method and its applications on downstream tasks. Figure 2 shows their workflows. For example, in the pre-training stage, given the word "happy" and its sentiment orientation "1" in a sentiment lexicon, we assign "happy" a randomly initialized sentiment mapping vector ∇_{happy}. Then, we add the word embedding of "happy" to the sentiment mapping vector and obtain a sentiment-enhanced word embedding. We feed it to a fully connected layer, predicting the probability of the "happy" sentiment label. By making a loss between the predicted sentiment label and the gold standard (i.e., 1), we can train ∇_{happy} by backpropagation. The ∇_{happy} will be the unique sentiment mapping vector for "happy." In the application stage, we add the well-trained ∇_{happy} and the original word embedding of "happy"; we obtain the sentiment-enhanced embedding of "happy" and apply the sentiment-enhanced embedding into the embedding layer of the downstream tasks.

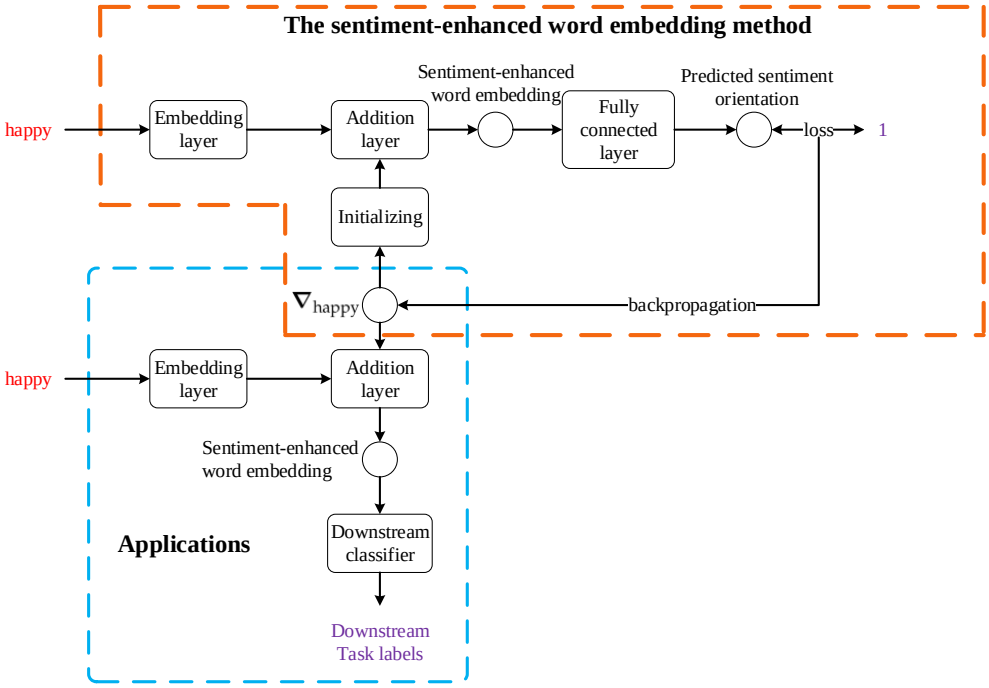


Figure 2. The workflow of S-EWE method and its applications on the downstream tasks. The orange dashed box indicates the stage in which we train the S-EWE embeddings, and the blue dashed box represents the stage in which we apply the embeddings to downstream tasks. The red texts are the inputs, and the purple texts are the outputs.

3.1. The Sentiment-Enhanced Word Embedding Method

The S-EWE method aims to establish the mapping relationship between words in sentiment lexicons and their sentiment orientations to obtain trained sentiment-enhanced word embeddings for downstream tasks. When we convert one word to a word embedding, the dimensions of the word embedding are much larger than those of the sentiment orientation, and they belong to two different vector spaces. Therefore, we assign a sentiment mapping vector to the word embedding. This sentiment mapping vector can help word embedding find the mapping relationship between it and its sentiment orientation.

Let the $\mathcal{V}$ be a set of all words in a sentiment lexicon, and $\mathcal{Y}$ be a set of sentiment orientations of all words in the sentiment lexicon. Given a word $w_i \in \mathcal{V}$, its word embedding $\boldsymbol{w}_i$, and its sentiment orientation $y_i \in \mathcal{Y}$, we assign a sentiment mapping vector $\boldsymbol{\nabla}_i$ to the word embedding $\boldsymbol{w}_i$. Then, the goal of the S-EWE method is to solve the mapping relationship function among w_i, $\boldsymbol{w}_i$ and $\boldsymbol{\nabla}_i$, denoted by $f(y_i|w_i, \boldsymbol{\nabla}_i)$. Figure 3 shows the running

process of the S-EWE method. The method implements $f(y_i|w_i, \nabla_i)$ by four modules, including the embedding layer, addition layer, MLP layer, and loss function as follows.

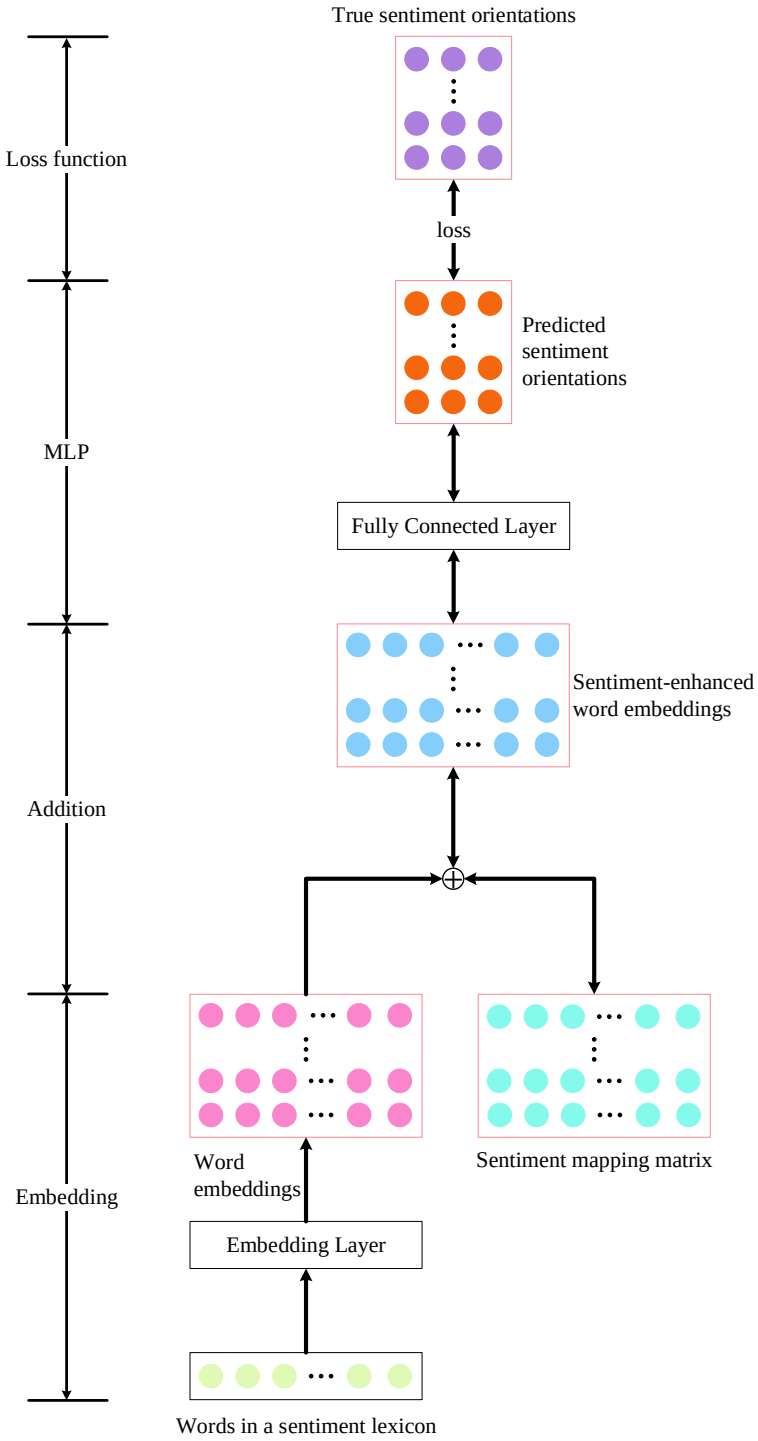


Figure 3. The training process diagram of the S-EWEs. The one-way arrow indicates forward propagation, and the two-way arrows represent forward and backward propagation.

3.1.1. Embedding Layer

The embedding layer converts words to word embeddings, which is the first step in finding the relationship between w_i and y_i. Let sentiment words $w = \{w_1, w_2, \ldots, w_n\}$ where $w_i \in \mathcal{V}$, and n is the total number of sentiment words, and their sentiment ori-

entations $Y = \{y_1, y_2, \ldots, y_n\} \in \mathcal{Y}$. Then, we first turn words into word embeddings as follows.

$$\boldsymbol{w} = \text{embed}(w) = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_n\}, \tag{1}$$

where $\boldsymbol{w}_i \in \mathbb{R}^{ed}$ is the word embedding of the word w_i (ed is the dimension of the embedding size), and embed$(\cdot)$ is the function that turns words into their word embeddings.

3.1.2. Addition Layer

The addition layer implements simple addition between vectors. We define a sentiment mapping matrix $\boldsymbol{\mathcal{R}} = \{\boldsymbol{\nabla}_1, \boldsymbol{\nabla}_2, \ldots, \boldsymbol{\nabla}_n\}$, where $\boldsymbol{\nabla}_i \in \mathbb{R}^{ed}$ is the mapping relationship vector between $\boldsymbol{w}_i$ and y_i. Let $\boldsymbol{E} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_n\}$ be the sentiment-enhanced word embeddings of sentiment words $w = \{w_1, w_2, \ldots, w_n\}$, where $\boldsymbol{e}_i \in \mathbb{R}^{ed}$. Then, by adding word embedding $\boldsymbol{w}_i$ to its sentiment mapping vector $\boldsymbol{\nabla}_i$, we can obtain the sentiment-enhanced word embedding $\boldsymbol{e}_i$ of the word w_i as follows.

$$\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{\nabla}_i. \tag{2}$$

Then, the sentiment-enhanced word embedding of $\boldsymbol{w}$ is as follows.

$$\boldsymbol{E} = \boldsymbol{w} + \boldsymbol{\mathcal{R}}. \tag{3}$$

3.1.3. MLP Layer

The MLP layer contains one fully connected network. It will reduce the dimensions of sentiment-enhanced word embeddings and map them and their sentiment orientation labels into the same vector space. Considering that the addition of vectors does not change the vectors' dimensions, the dimensions of $\boldsymbol{e}_i$ are still much more than those of y_i. We use one MLP to reduce the dimensionality of the sentiment-enhanced word vector. The MLP will map the sentiment-enhanced word embedding and label to the same vector space [38]. The predicted sentiment orientation label of the word w_i, denoted by $\hat{y}_i$, is as follows.

$$\hat{y}_i = \text{softmax}(\boldsymbol{W}\boldsymbol{e}_i + \boldsymbol{b}), \tag{4}$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weight matrix and bias of the output layer, respectively.

3.1.4. Loss Function

The loss function measures the difference between true sentiment orientation labels in a sentiment lexicon and predicted sentiment orientation labels of words at the final module of the S-EWE method. We use the cross-entropy function

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)], \tag{5}$$

to calculate the loss, where N is the total number of samples.

In order to better learn the mapping relationship between original word embeddings and their true sentiment orientation labels in the S-EWE method, we freeze the word embedding layer. All word embeddings and true sentiment orientation labels are fixed in this case. The parameters W, b, and sentiment mapping matrix are trainable. By backpropagation, the method can better learn the sentiment mapping matrix and find the mapping relationship between original word embeddings and their corresponding sentiment orientations. After training, we can obtain the well-trained sentiment mapping matrix, denoted as $\boldsymbol{\mathcal{R}}^t = \{\boldsymbol{\nabla}_1^t, \boldsymbol{\nabla}_2^t, \ldots, \boldsymbol{\nabla}_n^t\}$. Finally, the well-trained sentiment-enhanced word embeddings, denoted by $\boldsymbol{E}^t = \{\boldsymbol{e}_1^t, \boldsymbol{e}_2^t, \cdots, \boldsymbol{e}_n^t\}$, are obtained as follows.

$$\boldsymbol{E}^t = \boldsymbol{w} + \boldsymbol{\mathcal{R}}^t. \tag{6}$$

The detailed running procedure of the S-EWE method is described in Algorithm 1.

Algorithm 1 The S-EWE algorithm.

Input: Set of words in a sentiment lexicon $w = \{w_1, w_2, \ldots, w_n\}$, set of sentiment orientations in the sentiment lexicon Y, number of epochs denoted by *epochs*, batches denoted by *batch*

Output: Well-trained sentiment-enhanced word embeddings $\boldsymbol{E}^t$

Initialization $\boldsymbol{\mathcal{R}} \leftarrow$ xavier Gaussian initialization
$\boldsymbol{w} \leftarrow \text{embed}(w)$ // Equation (1)
$i \leftarrow 0$
while $i < epochs$ **do**
  for $(\boldsymbol{w}, \boldsymbol{y}) \in batch$ **do**
    $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{\nabla}_i$ // Equation (2)
    $\hat{\boldsymbol{y}} \leftarrow \text{Softmax}(\boldsymbol{W}\boldsymbol{e}_i + \boldsymbol{b})$ // Equation (4)
    $\mathcal{L} \leftarrow -\frac{1}{\text{len}(batch)} \sum_{i=1}^{\text{len}(batch)} [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$ //Equation (5)
    Backpropagation
    Optimize parameters
  end for
  $i \leftarrow i + 1$
  Obtain well-trained sentiment-enhanced word embeddings $\boldsymbol{\mathcal{R}^t}$
end while
return well-trained sentiment mapping matrix $\boldsymbol{E}^t = \boldsymbol{w} + \boldsymbol{\mathcal{R}^t}$ //Equation (6)

3.2. Applications of the Sentiment-Enhanced Word Embedding Method on Downstream Tasks

Since the S-EWE method needs to work with a sentiment lexicon, we need a robust sentiment lexicon. We selected the extended version of Affective Norms of English Words (E-ANEW) [50] and the Subjectivity Clue Lexicon [51] as our base sentiment lexicons, for they are well-known English sentiment lexicons. For the E-ANEW [50], we define sentiment words with sentiment intensity between $[1, 5)$ as negative, sentiment words with sentiment intensity of 5 as neutral, and sentiment words with sentiment intensity between $(5, 9]$ as positive. For the Subjectivity Clue Lexicon [51], we directly extract the sentiment orientation (positive, neutral, negative).

To improve the robustness of these two lexicons, we adopted the following four rules to integrate them: (1) if a word appears in only one lexicon, then the word is directly added into the fused lexicon; (2) if a word has the same sentiment orientation in the two lexicons, then the word is directly added into the fused lexicon; (3) if the sentiment orientation of a word is neutral in one lexicon and the sentiment orientation is not neutral in another lexicon, then the word is assigned with non-neutral sentiment orientation and added into the fused lexicon; and (4) if a word has opposite sentiment orientations (positive and negative) in the two lexicons, then the word is discarded and does not belong to the fused lexicon. After finishing the above four rules, we can obtain *the fused lexicon*. Table 1 shows the details of the E-ANEW, Subjectivity Clue Lexicon, and fused lexicon, respectively.

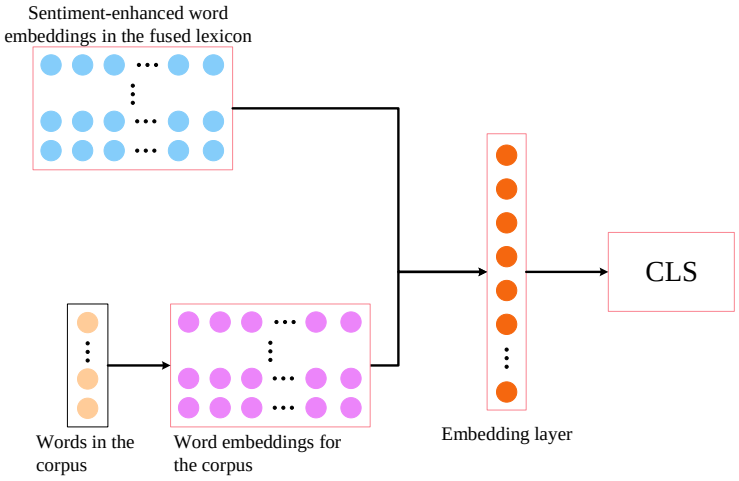
Table 1. The details of the sentiment lexicons.

	E-ANEW	Subjectivity Clue Lexicon	Fused Lexicon
Total number of words	13,915	6869	17,293
The number of positive words	7761	2296	8672
The number of negative words	5945	4149	8216
The number of neutral words	209	424	405

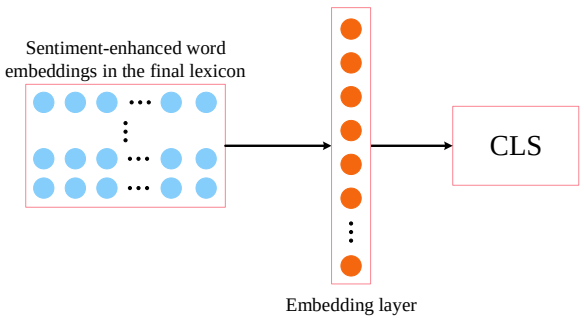
The words in the sentiment lexicon may not all appear in the corpus in most cases. Meanwhile, the S-EWE method will waste some storage space to train the words in the lexicon and may not obtain good sentiment-enhanced word embeddings. Inspired by sentiment-aware word embeddings (SAWE) [14], we extract all words from the corpus and

build a *final lexicon* by the following two rules. (1) If a word appears in both the corpus and the fused lexicon, we directly add it and its corresponding sentiment orientation in the fused lexicon to the final lexicon. (2) If a word appears in the corpus but not in the fused lexicon, we add it to the final lexicon with the neutral sentiment.

Based on the fused and final lexicons, the sentiment-enhanced word embeddings are applied to downstream tasks for sentiment analysis, as shown in Figure 4. When the sentiment-enhanced word embeddings are trained by the S-EWE method on the fused lexicon (respectively, the final lexicon), we denote the well-trained embeddings by S-EWE$_f$ (respectively, S-EWE$_c$). Figure 4 shows the applications of the sentiment-enhanced word embeddings trained by the S-EWE method based on the final and fused lexicons for downstream tasks. Figure 4a shows how we apply S-EWE$_f$ to downstream tasks, and Figure 4b represents how we apply S-EWE$_c$ to downstream tasks. When the fused lexicon does not contain some words in the corpus, we adopt the following two methods to get the embedding layer of the downstream sentiment analysis models (Figure 4a). (1) If a word is in the fused lexicon, we input the well-trained sentiment-enhanced word embedding by the S-EWE method in the fused lexicon as the word embedding with sentiment information into the embedding layer of the downstream tasks. (2) If a word is not in the fused lexicon, we use the original word embedding as the word embedding with sentiment information into the embedding layer of the downstream tasks. When the final lexicon contains all words in the corpus (Figure 4b), we can directly input the well-trained sentiment-enhanced word embeddings by the S-EWE method in the final lexicon as the word embedding with sentiment information into the embedding layer of the downstream tasks.



(a) Sentiment-enhanced word embeddings in the fused lexicon for downstream task



(b) Sentiment-enhanced word embeddings in the final lexicon for downstream task

Figure 4. The applications of the S-EWE method with the fused and final lexicons on downstream tasks.

4. Experiment

4.1. Datasets and Evaluation Metrics

The SemEval-2013 [27] and SST-2 (the preprocessed SST-2 dataset was selected as an experimental dataset; the preprocessed data are at https://github.com/clairett/pytorch-sentiment-classification, accessed on 26 November 2021) [28] datasets were selected as experimental datasets. The SemEval-2013 dataset is a Tweet sentiment analysis dataset containing five categories of labels (positive, negative, neutral, objective, and objective-or-neutral). Our experiment only considered the positive and negative data by SemEval-2013 (binary). The SST-2 dataset contains movie reviews containing two labels: (1) positive and negative, and (2) very negative, negative, neutral, positive, and very positive. From this dataset, our experiment only used the positive and negative data. Tables 2 and 3 show the details of all the datasets.

Table 2. The details of SemEval-2013 and SST-2 datasets.

	SemEval-2013 (Binary) [27]			**SST-2 [28]**		
	Positive	**Negative**	**Total**	**Positive**	**Negative**	**Total**
Train	2974	1159	4133	3610	3310	6920
Develop	483	280	763	444	428	872
Test	1281	472	1753	912	909	1821

Table 3. The numbers of sentiment words in SemEval-2013 and SST-2.

Datasets	**Total Words**	**Positive Words**	**Negative Words**	**Neutral Words**
SemEval-2013	15,133	2755	1412	10,966
SST-2	16,008	4001	2994	9013

The accuracy and macro-F1 score [14] were adopted to evaluate the performance of the proposed S-EWE method. Let TP (resp. TN) be the number of samples whose true labels and model prediction labels are both positive (resp. negative), and let FN (resp. FP) be the number of samples whose true labels are positive (resp. negative) and model prediction labels are negative (resp. positive). Accuracy and macro-F1 score are defined as

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{7}$$

and

$$\text{Macro-F1} = \frac{1}{k}\sum_{i=1}^{k} F1_i, \tag{8}$$

where $P_i = \frac{TP_i}{TP_i+FP_i}$, $R_i = \frac{TP_i}{TP_i+FN_i}$ and $\text{F1}_i = \frac{2P_iR_i}{P_i+R_i}$ stand for the precision, recall, and F1 scores of the category i; and k is the total number of categories.

4.2. Comparison Experimental Models and Their Training Details

TextCNN [22], TextRNN [23], TextRCNN (our main reference codes come from: https://github.com/649453932/Chinese-Text-Classification-Pytorch, accessed on 5 December 2021) [24], ABCDM [25], and CNN-LSTM [26] were selected as sentiment classification models to verify the effectiveness of trained sentiment-enhanced word embeddings by the S-EWE method in fused and final lexicons.

TextCNN. TextCNN [22] convolves word vectors in the convolution kernel. It obtains the feature maps of these convolution results through max pooling.

TextRNN. TextRNN [23] is a bidirectional LSTM. It inputs word embeddings and outputs the hidden state of each word.

TextRCNN. TextRCNN [24] uses LSTM (resp. BiLSTM) to obtain the forward (resp. bidirectional) hidden state. It applies max pooling to get sentence embeddings.

ABCDM. ABCDM [25] adopts BiLSTM and BiGRU to encode one sentence to obtain long-term and short-term hidden states. Then, it uses two parallel attention layers to obtain the attention score of each hidden state. Finally, ABCDM applies 1D-CNN, 1D-max pooling, and 1D-average pooling to get the final sentence embedding.

CNN-LSTM. CNN-LSTM [26] uses TextCNN [22] to obtain local sentence embedding, and adopts LSTM to get the long-term information.

In these sentiment classification models, Adam [52] optimizer is regarded as the optimizer. The cross-entropy function is selected as the loss function. The classical feature-based word embedding methods, including Word2Vec [1] and GloVe [2], are used to pre-train word embeddings. Specifically, GoogleNews-vectors-negative300 (https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?resourcekey=0-wjGZdNAUop6WykTtMip30g, accessed on 5 November 2021 and glove.840B.300d (https://nlp.stanford.edu/data/glove.840B.300d.zip, accessed on 10 December 2021) were adopted. The embedding dimensions of Word2Vec and GloVe were 300. All parameters of the S-EWE method and downstream task models are shown in Table 4.

Table 4. Parameter details of different models.

	S-EWE	TextCNN	TextRNN	TextRCNN	TextBiRCNN	ABCDM	CNN-LSTM
epochs	200	40	40	40	40	100	100
batch size	64	64	64	128	128	64	64
dropout	-	0.5	0.5	1.0	1.0	0.5	0.5
learning rate	1×10^{-3}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
early stop	512	256	256	512	512	512	512
filter sizes	-	(3, 4, 5)	-	-	-	(4, 6)	3
number of filters	-	100	-	-	-	32	600
hidden sizes	-	-	128	256	256	128	600
hidden layers	-	-	1	1	1	1	1

4.3. Experimental Results

Let the sentiment classification (SC) models enhanced by S-EWE$_c$ and S-EWE$_f$ be denoted by SC$_{\text{S-EWE}_c}$ and SC$_{\text{S-EWE}_f}$, respectively. When Word2Vec was used to pre-train word embeddings for sentiment classification tasks on SemEval-2013 [27] and SST-2 [28] datasets, the accuracies and macro-F1 scores under the sentiment classification models TextCNN, TextRNN, TextRCNN, TextBiRCNN, ABCDM, and CNN-LSTM with and without S-EWE$_c$ and S-EWE$_f$ are shown in Table 5. We conclude that: (1) the sentiment classification models with Word2Vec enhanced by S-EWE$_f$ and S-EWE$_c$ had higher accuracies and macro-F1 than the original Word2Vec except for TextRNN and CNN-LSTM; (2) in most cases, the sentiment classification models with Word2Vec enhanced by S-EWE$_c$ achieved the best accuracies and macro-F1 scores on two datasets; (3) TextRNN with Word2Vec enhanced by S-EWE$_f$ and S-EWE$_c$ achieved the same accuracies and macro-F1 scores of the TextRNN with original Word2Vec—that is, the S-EWE method did not improve the performance of TextRNN on the two datasets; and (4) on the two datasets, the variance of the results obtained by the model using the proposed method was minor, indicating that using S-EWE method can make the model more stable.

When GloVe is used to pre-train word embeddings for sentiment classification tasks on SemEval-2013 [27] and SST-2 [28] datasets, the accuracy and maro-F1 scores under sentiment classification models, including TextCNN, TextRNN, TextRCNN, TextBiRCNN, ABCDM, and CNN-LSTM, enhanced by S-EWE$_c$ and S-EWE$_f$ are shown in Table 6. Compared with these results, we can conclude that: (1) the sentiment classification models with GloVe enhanced by S-EWE$_f$ and S-EWE$_c$ had higher accuracies and macro-F1 scores than the original GloVe except for TextRNN; (2) in most cases, the sentiment classification models with GloVe enhanced by S-EWE$_c$ achieved the best accuracies and macro-F1 scores on two datasets; (3) TextRNN is not trainable on these two datasets with the proposed method; and (4) models using our method achieved more robust results on both datasets.

Table 5. The accuracies and macro-F1 scores of different models using Word2Vec with and without S-EWE$_c$ and S-EWE$_f$ on SemEval-2013 and SST-2 datasets (%). For each model, we ran it five times to achieve five accuracy values and macro-F1 values, respectively. Then, we took their average accuracies and macro-F1 values as the final results. The value in parentheses is the variance of the five values for each model. The bold text represents the best result achieved by each model, and the underlined text stands for the lowest variance achieved by each model.

Models	SemEval-2013		SST-2	
	Accuracy	**Macro-F1**	**Accuracy**	**Macro-F1**
TextCNN [22]	80.02 (0.34)	73.42 (1.22)	79.81 (0.83)	79.81 (0.83)
TextCNN$_{\text{S-E-WE}_f}$	**80.55** (0.17)	74.13 (0.52)	**80.89** (0.15)	**80.88** (0.15)
TextCNN$_{\text{S-EWE}_c}$	80.52 (0.23)	**74.55** (0.71)	80.69 (0.12)	80.68 (0.12)
TextRNN [23]	73.07 (0)	42.22 (0)	49.92 (0)	33.30 (0)
TextRNN$_{\text{S-EWE}_f}$	73.07 (0)	42.22 (0)	49.92 (0)	33.30 (0)
TextRNN$_{\text{S-EWE}_c}$	73.07 (0)	42.22 (0)	49.92 (0)	33.30 (0)
TextRCNN [24]	73.82 (0.09)	60.31 (44.45)	73.64 (1.13)	73.47 (1.34)
TextRCNN$_{\text{S-EWE}_f}$	74.87 (1.73)	**65.38** (2.85)	75.65 (1.05)	75.61 (1.1)
TextRCNN$_{\text{S-EWE}_c}$	**75.42** (0.19)	64.81 (22.84)	**75.72** (0.15)	**75.66** (0.13)
TextBiRCNN [24]	74.07 (1.24)	68.57 (1.14)	74.15 (0.74)	74.08 (0.75)
TextBiRCNN$_{\text{S-EWE}_f}$	**76.67** (1.05)	69.95 (1.03)	75.97 (0.33)	75.95 (0.34)
TextBiRCNN$_{\text{S-EWE}_c}$	76.13 (0.55)	**71.22** (1.89)	**76.17** (0.24)	**76.16** (0.24)
ABCDM [25]	74.85 (2.98)	69.36 (0.61)	74.26 (0.54)	74.22 (0.59)
ABCDM$_{\text{S-EWE}_f}$	**76.08** (0.34)	68.84 (0.52)	75.15 (0.13)	75.10 (0.12)
ABCDM$_{\text{S-EWE}_c}$	75.09 (0.32)	**69.58** (1.21)	**75.51** (0.06)	**75.47** (0.06)
CNN-LSTM [26]	**76.39** (2.54)	68.43 (7.85)	75.75 (0.35)	75.73 (0.33)
CNN-LSTM$_{\text{S-EWE}_f}$	75.48 (3.17)	68.47 (0.25)	76.06 (0.12)	76.01 (0.11)
CNN-LSTM$_{\text{S-EWE}_c}$	76.19 (1.35)	**69.62** (0.52)	**76.40** (0.36)	**76.38** (0.36)

Table 6. The accuracies and macro-F1 scores of different models using GloVe with/without S-EWE$_c$ and S-EWE$_f$ on SemEval-2013 and SST-2 datasets (%). The accuracies and macro-F1 values were obtained in the same way as those in Table 5. Moreover, the values in parentheses, bold text, and underlined text represent the common meanings in Table 5.

Models	SemEval-2013		SST-2	
	Accuracy	**Macro-F1**	**Accuracy**	**Macro-F1**
TextCNN [22]	80.07 (0.02)	73.17 (0.02)	79.88 (0.31)	79.88 (0.31)
TextCNN$_{\text{S-EWE}_f}$	80.11 (0.02)	73.27 (0.03)	79.99 (0.01)	79.99 (0.01)
TextCNN$_{\text{S-EWE}_c}$	**80.62** (0.05)	**73.68** (0.19)	**80.14** (0.16)	**80.14** (0.16)
TextRNN [23]	73.07 (0)	42.22 (0)	49.92 (0)	33.30 (0)
TextRNN$_{\text{S-EWE}_f}$	73.07 (0)	42.22 (0)	49.92 (0)	33.30 (0)
TextRNN$_{\text{S-EWE}_c}$	73.07 (0)	42.22 (0)	49.92 (0)	33.30 (0)
TextRCNN [24]	72.62 (1.00)	64.63 (10.90)	72.56 (3.51)	72.45 (4.00)
TextRCNN$_{\text{S-EWE}_f}$	**74.03** (0.78)	66.40 (10.34)	74.83 (0.30)	74.78 (0.35)
TextRCNN$_{\text{S-EWE}_c}$	73.25 (2.26)	**67.40** (1.38)	**75.65** (0.63)	**75.63** (0.63)
TextBiRCNN [24]	73.66 (2.57)	66.37 (1.26)	73.32 (0.83)	73.29 (0.83)
TextBiRCNN$_{\text{S-EWE}_f}$	74.92 (1.13)	66.81 (1.15)	75.12 (0.85)	75.10 (0.9)
TextBiRCNN$_{\text{S-EWE}_c}$	**75.70** (0.26)	**67.50** (2.58)	**75.29** (0.19)	**75.25** (0.19)
ABCDM [25]	73.77 (5.42)	64.29 (11.64)	73.48 (0.71)	73.43 (0.66)
ABCDM$_{\text{S-EWE}_f}$	**74.76** (0.68)	**67.56** (1.50)	**74.97** (1.45)	**74.94** (1.46)
ABCDM$_{\text{S-EWE}_c}$	73.94 (1.72)	67.10 (0.27)	74.31 (0.17)	74.28 (0.15)
CNN-LSTM [26]	74.82 (3.77)	67.84 (2.02)	71.65 (3.74)	71.56 (4.30)
CNN-LSTM$_{\text{S-EWE}_f}$	76.85 (0.39)	**68.91** (1.87)	72.38 (0.27)	72.35 (0.26)
CNN-LSTM$_{\text{S-EWE}_c}$	**76.95** (0.49)	68.72 (1.72)	**72.56** (0.06)	**72.51** (0.07)

As shown in Tables 5 and 6, TextCNN, TextRCNN, TextBiRCNN, ABCDM, and CNN-LSTM with Word2Vec or GloVe, enhanced by S-EWE$_c$ and S-EWE$_f$, achieved better classification performances than these models without the sentiment-enhanced word embeddings on the two datasets. In particular, these models with S-EWE$_c$ obtained better classification performances than those with S-EWE$_f$. TextRNN with Word2Vec or GloVe enhanced by S-EWE$_c$ and S-EWE$_f$ kept the same classification performance as the model without the sentiment-enhanced word embedding on the two datasets. Meanwhile, models using S-EWE$_c$ and S-EWE$_f$ can achieve more robust results on the two datasets.

4.4. Analysis of Convergence Time for Downstream Tasks

Under TextCNN, TextRNN, TextRCNN, TextBiRCNN, ABCDM, and CNN-LSTM using Word2Vec and GloVe with or without S-EWE$_c$ and S-EWE$_f$ on SemEval-2013 and SST-2 datasets, their convergence times were analyzed on one NVIDIA 2080Ti (12GB), as shown in Table 7. For all baseline models using Word2Vec enhanced by S-EWEs on the SemEval-2013 dataset, the convergence times of TextCNN$_{\text{S-EWE}_c}$, TextRCNN$_{\text{S-EWE}_c}$, and TextBiRCNN$_{\text{S-EWE}_c}$ were on average increased by 22.56% over their original models; however, the convergence times of TextRNN$_{\text{S-EWE}_c}$, ABCDM$_{\text{S-EWE}_c}$, and CNN-LSTM$_{\text{S-EWE}_c}$ on average decreased by 31.16%. Meanwhile, for all baseline models using Word2Vec enhanced by S-EWEs on the SST-2 dataset, the convergence time of CNN-LSTM$_{\text{S-EWE}_c}$ was the most significant increase, by 440.32% over CNN-LSTM; the convergence time of TextRNN$_{\text{S-EWE}_c}$ decreased the convergence time of TextRNN by 0.35%; and the convergence times of TextCNN$_{\text{S-EWE}_c}$, TextRCNN$_{\text{S-EWE}_c}$, TextBiRCNN$_{\text{S-EWE}_c}$, and ABCDM$_{\text{S-EWE}_f}$ were on average increased by 21.98% of their original models.

For all baseline models using GloVe enhanced by S-EWEs on the SemEval-2013 dataset, the convergence times of TextCNN$_{\text{S-EWE}_c}$, TextRCNN$_{\text{S-EWE}_c}$, TextBiRCNN$_{\text{S-EWE}_c}$, and ABCDM$_{\text{S-EWE}_c}$ were on average increased by 50.17% compared to their original models. The convergence time of CNN-LSTM$_{\text{S-EWE}_c}$ was decreased by 26.24% compared to CNN-LSTM. Meanwhile, using GloVe enhanced by S-EWEs on the SST-2 dataset, the convergence times of TextBiRCNN$_{\text{S-EWE}_c}$ and ABCDM$_{\text{S-EWE}_c}$ were on average increased by 19.67% compared to their original models; the convergence times of TextCNN$_{\text{S-EWE}_c}$, TextRCNN$_{\text{S-EWE}_c}$, and CNN-LSTM$_{\text{S-EWE}_c}$ were on average decreased by 25.34% compared to their original models.

In all baseline models with the S-EWE method, the convergence time of ABCDM$_{\text{S-EWE}_f}$ was significantly decreased by 51.21% compared to ABCDM on the SemEval-2013 dataset, and the convergence time of CNN-LSTM$_{\text{S-EWE}_C}$ vastly decreased by 41.34% compared to CNN-LSTM on the SST-2 dataset.

Based on the performances of various sentiment classification models and models' convergence times, we conclude that the proposed S-EWE method can enhance the ability of sentiment classification on SemEval-2013 and SST-2 datasets.

Table 7. Comparisons with the convergence times of TextCNN, TextRNN, TextRCNN, TextBiRCNN, ABCDM, and CNN-LSTM using Word2Vec and GloVe with or without S-EWE$_c$ and S-EWE$_f$ on two datasets. The contents in parentheses indicate the increased or decreased convergence speeds compared to the original models .

Input Embedding	Model \ Dataset	SemEval-2013	SST-2
Word2Vec	TextCNN [22]	19.36	18.54
	TextCNN$_{\text{S-EWE}_f}$	21.08 (8.88% ↑)	28.76 (55.12% ↑)
	TextCNN$_{\text{S-EWE}_c}$	21.50 (11.05% ↑)	25.36 (36.79% ↑)
	TextRNN [23]	67.50	54.49
	TextRNN$_{\text{S-EWE}_f}$	68.86 (2.01% ↑)	54.49 (0%)
	TextRNN$_{\text{S-EWE}_c}$	64.71 (−4.13% ↓)	54.30 (−0.35% ↓)
	TextRCNN [24]	392.75	459.07
	TextRCNN$_{\text{S-EWE}_f}$	526.43 (34.04% ↑)	346.01 (-24.63% ↓)
	TextRCNN$_{\text{S-EWE}_c}$	536.14 (36.51% ↑)	499.65 (8.84% ↑)
	TextBiRCNN [24]	770.44	647.97
	TextBiRCNN$_{\text{S-EWE}_f}$	825.24 (7.11% ↑)	647.19 (−0.12% ↓)
	TextBiRCNN$_{\text{S-EWE}_c}$	925.44 (20.12% ↑)	715.88 (10.48% ↑)
	ABCDM [25]	97.04	46.77
	ABCDM$_{\text{S-EWE}_f}$	47.35 (−**51.21**% ↓)	60.61 (29.59% ↑)
	ABCDM$_{\text{S-EWE}_c}$	53.70 (−44.66% ↓)	61.65% (31.82% ↑)
	CNN-LSTM [26]	190.94	71.09
	CNN-LSTM$_{\text{S-EWE}_f}$	174.97 (−8.36% ↓)	177.24 (149.32% ↑)
	CNN-LSTM$_{\text{S-EWE}_c}$	105.59 (−44.70% ↓)	384.11 (**440.32**% ↑)
GloVe	TextCNN [22]	18.12	34.44
	TextCNN$_{\text{S-EWE}_f}$	21.93 (21.03% ↑)	29.07 (−15.59% ↓)
	TextCNN$_{\text{S-EWE}_c}$	21.74 (19.98% ↑)	31.55 (−8.39% ↓)
	TextRNN [23]	66.32	55.65
	TextRNN$_{\text{S-EWE}_f}$	66.78 (0.69% ↑)	70.94 (27.48% ↑)
	TextRNN$_{\text{S-EWE}_c}$	66.63 (0.47% ↑)	69.94 (25.68% ↑)
	TextRCNN [24]	339.23	576.37
	TextRCNN$_{\text{S-EWE}_f}$	390.31 (15.06% ↑)	461.90 (−19.86% ↓)
	TextRCNN$_{\text{S-EWE}_c}$	437.42 (28.94% ↑)	424.92 (−26.28% ↓)
	TextBiRCNN [24]	822.67	719.04
	TextBiRCNN$_{\text{S-EWE}_f}$	822.35 (−0.04% ↓)	790.64 (9.96% ↑)
	TextBiRCNN$_{\text{S-EWE}_c}$	824.52 (0.22% ↑)	794.55 (10.50% ↑)
	ABCDM [25]	47.59	74.07
	ABCDM$_{\text{S-EWE}_f}$	64.50 (35.53% ↑)	78.82 (6.41% ↑)
	ABCDM$_{\text{S-EWE}_c}$	119.70 (151.52% ↑)	95.43% (28.84% ↑)
	CNN-LSTM [26]	164.84	189.07
	CNN-LSTM$_{\text{S-EWE}_f}$	159.77 (−3.08% ↓)	222.25 (17.55% ↑)
	CNN-LSTM$_{\text{S-EWE}_c}$	121.59 (−26.24% ↓)	110.91 (−41.34% ↓)

4.5. Comparisons with the Contextualized Word Embedding Models and the BERT with/without the S-EWEs

The contextualized word embedding models, including BERT [3] and CoSE-T [6], were selected to compare with the sentiment-enhanced word embedding method.

BERT [3]. It is a classic transformer-based [30] model. Its backbone is the encoder in a transformer. It adopts the masked language model (MLM) and next sentence prediction (NSP) as the pre-training objectives. When it is doing the downstream sentiment analysis task, the hidden state of the special token (CLS) it outputs will go through a pooler layer and output the classification result. We adopt the BERT base as our experimental model.

CoSE-T [6]. It has two BiGRU layers. In the pre-training stage, the target words, word sentiment, and sentence sentiment are predicted by training on the labeled sentiment corpus with a sentiment lexicon. In the fine-tuning stage, the pre-trained word embedding and the hidden states of the two BiGRUs are concatenated as the sentiment representation.

Finally, the representation will be fed into one fully connected layer with a softmax function to predict sentence sentiment.

BERT with S-EWEs . We extract the weights in the embedding layer of BERT and use S-EWE$_f$ and S-EWE$_c$ to enhance these weights. After enhancing the sentiment of the BERT embeddings, we replace these weights with the S-EWEs. Finally, we feed them into the BERT model.

Table 8 shows the accuracies and macro-F1 values of the BERT and CoSE-T models on SemEval-2013 and SST-2 datasets. From these results, we can observe that: (1) Contextualized word embedding can achieve better results than feature-based word embedding (shown in Table 5 and 6) in sentiment analysis tasks. Specifically, CoSE-T achieved 90.80% accuracy on the SemEval-2013 dataset and 89.80% accuracy on the SST-2 dataset; BERT achieved 90.80% accuracy and 88.88% macro-F1 on the SemEval-2013 dataset, and 90.38% accuracy and 90.38% macro-F1 on the SST-2 dataset. (2) BERT with S-EWE embeddings performs worse than BERT and CoSE-T without S-EWE embeddings. Specifically, on the SemEval-2013 dataset, BERT$_{\text{S-EWE}_f}$ was 9.14% less accurate than BERT and CoSE-T, and 11.34% lower than BERT in macro-F1; BERT$_{\text{S-EWE}_c}$ was 14.97% less accurate than BERT and CoSE-T and 20.21% lower than BERT in macro-F1. On the SST-2 dataset, BERT$_{\text{S-EWE}_f}$ was 10.39% lower than CoSE-T and 10.97% lower than BERT in accuracy, and 11.34% lower than BERT in macro-F1; BERT$_{\text{S-EWE}_c}$ was 17.14% lower than CoSE-T and 17.72% lower than BERT in accuracy, and 17.73% lower than BERT in macro-F1. From these observations, we can conclude that the S-EWE method is unsuitable for enhancing contextualized word embedding. The main reasons are that the S-EWE method only enhances the embedding layer of the model and has no effect on the model itself.

Table 8. Theaccuracies and macro-F1 scores of BERT (with and without S-EWE) and CoSE-T on SemEval-2013 and SST-2 datasets (%). † represents that the results are from [6].

Model	SemEval-2013		SST-2	
	Accuracy	Macro-F1	Accuracy	Macro-F1
CoSE-T †	90.80	-	89.80	-
BERT	90.80	88.88	90.38	90.38
BERT$_{\text{S-EWE}_f}$	81.66	77.54	79.41	79.40
BERT$_{\text{S-EWE}_c}$	75.83	68.67	72.66	72.65

5. Conclusions

This paper proposes a sentiment enhancement method, i.e., the sentiment-enhanced word embedding method. This model finds the relationship between the words in the sentiment lexicon and their corresponding sentiment orientations. After training the sentiment mapping matrix, this matrix and word embeddings are fused as sentiment-enhanced word embeddings. Then, the sentiment-enhanced word embeddings are fed into sentiment classification models to classify the sentiment orientations of sentences on SemEval-2013 and SST-2 datasets. Experimental results show that these models using Word2Vec and GloVe enhanced by the sentiment-enhanced word embeddings perform better than those with original Word2Vec and GloVe embeddings. Moreover, the convergence times of the models using Word2Vec and GloVe enhanced by the sentiment-enhanced word embeddings were acceptable. Since the S-EWE method only enhances the embedding layer of the model and has no effect on the model itself, it does not effectively enhance contextualized word embedding.

In the future, there will be some sustainable research directions. (1) The proposed sentiment mapping embeddings could be considered external knowledge in other sentiment classification models. (2) Since sentiment orientations contain less information than sentiment intensity, sentiment intensity could be considered in the S-EWE method. (3) Since Chinese words contain one or more tokens, various Chinese sentiment-enhanced word embedding methods need further study by expanding the S-EWE method. (4) The

methods of injecting sentiment into contextualized word embeddings, such as BERT, need to be used in sentiment analysis.

Author Contributions: Methodology, software, data curation and writing—original draft, Q.L.; supervision, writing—review and editing, funding acquisition, and formal analysis, X.L.; funding acquisition, investigation, and validation, Y.D.; formal analysis and software, Y.F.; funding acquisition and formal analysis, X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the Sichuan Science and Technology Program (2022YFG0378, 2021YFQ0008), the National Natural Science Foundation of China (61802316, 61872298, 61902324) and the Innovation Fund of Postgraduate, Xihua University (grant number YCJJ2021025).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request. All source codes and experiment details are available at https://github.com/Balding-Lee/ESWV, accessed on 13 March 2022.

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

Acknowledgments: The authors would like to thank all editors and anonymous reviewers for their valuable comments and suggestions, which have significantly improved the quality and presentation of this paper.

References

1. 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 2013, Scottsdale, AZ, USA, 2–4 May 2013.
2. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
3. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MI, USA, 2–7 June 2019; pp. 4171–4186.
4. Chen, Q.; Li, C.; Li, W. Modeling language discrepancy for cross-lingual sentiment analysis. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, 6–10 November 2017; pp. 117–126.
5. Jain, D.K.; Boyapati, P.; Venkatesh, J.; Prakash, M. An intelligent cognitive-inspired computing with big data analytics framework for sentiment analysis and classification. *Inf. Process. Manag.* **2022**, *59*, 102758. [CrossRef]
6. Wang, J.; Zhang, Y.; Yu, L.; Zhang, X. Contextual sentiment embeddings via bi-directional GRU language model. *Knowl.-Based Syst.* **2021**, *235*, 107663. [CrossRef]
7. Wu, J.; Ye, C.; Zhou, H. BERT for sentiment classification in software engineering. In Proceedings of the International Conference on Service Science, ICSS 2021, Xi'an, China, 14–16 May 2021; pp. 115–121.
8. Zhao, A.; Yu, Y. Knowledge-enabled BERT for aspect-based sentiment analysis. *Knowl.-Based Syst.* **2021**, *227*, 107220. [CrossRef]
9. Chaudhary, A.; Sheikh, Z.; Anastasopoulos, A.; Neubig, G. Reducing confusion in active learning for part-of-speech tagging. *Trans. Assoc. Comput. Linguist.* **2021**, *9*, 1–16. [CrossRef]
10. Rodríguez, A.J.C.; Castro, D.C.; García, S.H. Noun-based attention mechanism for fine-grained named entity recognition. *Expert Syst. Appl.* **2022**, *193*, 116406. [CrossRef]
11. Li, W.; Li, Y.; Liu, W.; Wang, C. An influence maximization method based on crowd emotion under an emotion-based attribute social network. *Inf. Process. Manag.* **2022**, *59*, 102818. [CrossRef]
12. Wang, Y.; Du, Y.; Xie, C. Emotion role identification in social network. In Proceedings of the Knowledge Graph and Semantic Computing: Knowledge Graph and Cognitive Intelligence—5th China Conference, CCKS 2020, Nanchang, China, 12–15 November 2020; pp. 287–298.
13. Sitaula, C.; Basnet, A.; Mainali, A.; Shahi, T.B. Deep Learning-Based Methods for Sentiment Analysis on Nepali COVID-19-Related Tweets. *Comput. Intell. Neurosci.* **2021**, *2021*, 2158184. [CrossRef]
14. Naderalvojoud, B.; Sezer, E.A. Sentiment aware word embeddings using refinement and senti-contextualized learning approach. *Neurocomputing* **2020**, *405*, 149–160. [CrossRef]
15. Yu, L.; Wang, J.; Lai, K.R.; Zhang, X. Refining word embeddings using intensity scores for sentiment analysis. *IEEE/ACM Trans. Audio Speech Language Process.* **2018**, *26*, 671–681. [CrossRef]
16. Ekman, P. An argument for basic emotions. *Cogn. Emot.* **1992**, *6*, 169–200. [CrossRef]
17. Xu, L.; Lin, H.; Pan, Y.; Ren, H.; Chen, J. Constructing the affective lexicon ontology. *J. China Soc. Sci. Tech. Inform.* **2008**, *27*, 180–185.

18. Demszky, D.; Movshovitz-Attias, D.; Ko, J.; Cowen, A.S.; Nemade, G.; Ravi, S. Goemotions: A dataset of fine-grained emotions. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020; pp. 4040–4054.
19. Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. In Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, NIPS 2013, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 2787–2795.
20. Grill, J.-B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.H.; Buchatskaya, E.; Doersch, C.; Pires, B.A.; Guo, Z.D.; Azar, M.G.; et al. Bootstrap your own latent-a new approach to self-supervised learning. In Proceedings of the Advances in Neural Information Processing Systems 2020, NIPS 2020, Virtual, 6–10 December 2020; pp. 21271–21284.
21. Li, Z.; Zou, Y.; Zhang, C.; Wei, Z. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event, 7–11 November 2021; pp. 246–256.
22. Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, Doha, Qatar, 26–28 October 2014; pp. 1746–1751.
23. Liu, P.; Qiu, X.; Huang, X. Recurrent neural network for text classification with multi-task learning. In Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9–15 July 2016; pp. 2873–2879.
24. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. In Proceedings of the 29th AAAI Conference on Artificial Intelligence, AAAI 2015, Austin, TX, USA, 25–30 January 2015; pp. 2267–2273.
25. Basiri, M.E.; Nemati, S.; Abdar, M.; Cambria, E.; Acharya, U.R. ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. *Future Gener. Comput. Syst.* **2021**, *115*, 279–294. [CrossRef]
26. Ankita; Rani, S.; Bashir, A.K.; Alhudhaif, A.; Koundal, D.; Gunduz, S.E. An efficient CNN-LSTM model for sentiment detection in #BlackLivesMatter. *Expert Syst. Appl.* **2022**, *193*, 116256.
27. Nakov, P.; Rosenthal, S.; Kozareva, Z.; Stoyanov, V.; Ritter, A.; Wilson, T. Semeval-2013 task 2: Sentiment analysis in twitter. In Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2013, , Atlanta, GA, USA, 14–15 June 2013, pp. 312–320.
28. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.Y.; Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.
29. Peters, M.E.; Ammar, W.; Bhagavatula, C.; Power, R. Semi-supervised sequence tagging with bidirectional language models. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1756–1765.
30. 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 30: Annual Conference on Neural Information Processing Systems, NIPS 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
31. Liu, C.; Hsu, T.; Chuang, Y.; Li, C.; Lee, H. Language representation in multilingual BERT and its applications to improve cross-lingual generalization. *arXiv* **2021**, arXiv:2010.10041.
32. Singh, L.G.; Mitra, A.; Singh, S.R. Sentiment analysis of tweets using heterogeneous multi-layer network representation and embedding. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Honolulu, HI, USA, 16–18 November 2020; pp. 8932–8946.
33. Moghadasi, M.N.; Zhuang, Y. Sent2vec: A new sentence embedding representation with sentimental semantic. In Proceedings of the IEEE International Conference on Big Data, Xi'an, China, 10–13 December 2020; pp. 4672–4680.
34. Arora, S.; Liang, Y.; Ma, T. A simple but tough-to-beat baseline for sentence embeddings. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.
35. Arora, S.; Li, Y.; Liang, Y.; Ma, T.; Risteski, A. A latent variable model approach to PMI-based word embeddings. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 385–399. [CrossRef]
36. Liang, D.; Yi, B. Two-stage three-way enhanced technique for ensemble learning in inclusive policy text classification. *Inf. Sci.* **2021**, *547*, 271–288. [CrossRef]
37. Bai, R.; Huang, R.; Chen, Y.; Qin, Y. Deep multi-view document clustering with enhanced semantic embedding. *Inf. Sci.* **2021**, *564*, 273–287. [CrossRef]
38. Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; Liu, Q. ERNIE: Enhanced language representation with informative entities. In Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, 28 July–2 August 2019; pp. 1441–1451.
39. Wang, X.; Gao, T.; Zhu, Z.; Zhang, Z.; Liu, Z.; Li, J.; Tang, J. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Trans. Assoc. Comput. Linguist.* **2021**, *9*, 176–194. [CrossRef]
40. 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. *arXiv* **2019**, arXiv:1907.11692.
41. Saxena, A.; Chakrabarti, S.; Talukdar, P.P. Question answering over temporal knowledge graphs. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, Virtual Event, 1–6 August 2021; pp. 6663–6676.

42. Mihaylov, T.; Frank, A. Knowledgeable reader: enhancing cloze-style reading comprehension with external commonsense knowledge. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, 15–20 July 2018; pp. 821–832.
43. Zhong, W.; Tang, D.; Duan, N.; Zhou, M.; Wang, J.; Yin, J. Improving question answering by commonsense-based pre-trainin. In Proceedings of the 8th CCF International Conference Natural Language Processing and Chinese Computing NLPCC 2019, Dunhuang, China, 9–14 October 2019; Volume 11838, pp. 16–28.
44. Shi, L.; Wu, W.; Guo, W.; Hu, W.; Chen, J.; Zheng, W.; He, L. SENGR: Sentiment-Enhanced neural graph recommender. *Inf. Sci.* **2022**, *589*, 655–669. [CrossRef]
45. Hamilton, W.L.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, NIPS 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 1024–1034.
46. Gavilanes, M.F.; Costa-Montenegro, E.; García-Méndez, S.; González-Casta no, F.J.; Juncal-Martínez, J. Evaluation of online emoji description resources for sentiment analysis purposes. *Expert Syst. Appl.* **2021**, *184*, 115279. [CrossRef]
47. Gavilanes, M.F.; Juncal-Martínez, J.; García-Méndez, S.; Costa-Montenegro, E.; González-Casta no, F.J. Creating emoji lexica from unsupervised sentiment analysis of their descriptions. *Expert Syst. Appl.* **2018**, *103*, 74–91. [CrossRef]
48. Gavilanes, M.F.; Álvarez-López, T.; Juncal-Martínez, J.; Costa-Montenegro, E.; González-Casta no, F.J. Unsupervised method for sentiment analysis in online texts. *Expert Syst. Appl.* **2016**, *58*, 57–75. [CrossRef]
49. Wei, J.; Liao, J.; Yang, Z.; Wang, S.; Zhao, Q. Bilstm with multi-polarity orthogonal attention for implicit sentiment analysis. *Neurocomputing* **2020**, *383*, 165–173. [CrossRef]
50. Warriner, A.B.; Kuperman, V.; Brysbaert, M. Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behav. Res. Methods* **2013**, *45*, 1191–1207. [CrossRef] [PubMed]
51. Wilson, T.; Wiebe, J.; Hoffmann, P. Recognizing contextual polarity in phrase-level sentiment analysis. In Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, HLT/EMNLP 2005, Vancouver, BC, Canada, 6–8 October 2005; pp. 347–354.
52. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.