future internet

MDPI

Article

An Improved Approach for Text Sentiment Classification Based on a Deep Neural Network via a Sentiment Attention Mechanism

Wenkuan Li, Peiyu Liu *, Qiuyue Zhang and Wenfeng Liu

School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China; 2017020847@stu.sdnu.edu.cn (W.L.); 2018309063@stu.sdnu.edu.cn (Q.Z.); liuwenfeng@stu.sdnu.edu.cn (W.L.)

* Correspondence: liupy@sdnu.edu.cn; Tel.: +86-183-6613-2394

Received: 10 February 2019; Accepted: 1 April 2019; Published: 11 April 2019

Abstract: Text sentiment analysis is an important but challenging task. Remarkable success has been achieved along with the wide application of deep learning methods, but deep learning methods dealing with text sentiment classification tasks cannot fully exploit sentiment linguistic knowledge, which hinders the development of text sentiment analysis. In this paper, we propose a sentiment-feature-enhanced deep neural network (SDNN) to address the problem by integrating sentiment linguistic knowledge into a deep neural network via a sentiment attention mechanism. Specifically, first we introduce a novel sentiment attention mechanism to help select the crucial sentiment-word-relevant context words by leveraging the sentiment lexicon in an attention mechanism, which bridges the gap between traditional sentiment linguistic knowledge and current popular deep learning methods. Second, we develop an improved deep neural network to extract sequential correlation information and text local features by combining bidirectional gated recurrent units with a convolutional neural network, which further enhances the ability of comprehensive text representation learning. With this design, the SDNN model can generate a powerful semantic representation of text to improve the performance of text sentiment classification tasks. Extensive experiments were conducted to evaluate the effectiveness of the proposed SDNN model on two real-world datasets with a binary-sentiment-label and a multi-sentiment-label. The experimental results demonstrated that the SDNN achieved substantially better performance than the strong competitors for text sentiment classification tasks.

Keywords: deep learning; sentiment attention mechanism; bidirectional gated recurrent unit; convolutional neural network

1. Introduction

With the exponential growth of large collections of opinion-rich resources, sentiment classification [1] has been one of the most important tasks in natural language processing (NLP), which aims to automatically classify the sentiment polarity of a given text as negative, positive or more fine-grained classes. It can help companies to process and extract precious data from massive amounts of information, which contain great business value in brand monitoring, customer services, market research, politics, and social services. For example, tracking consumers' overall appreciation of a certain product can assist merchants in adjusting their marketing strategy.

A fundamental problem of NLP is text representation learning, which is encoding the text into continuous vectors by constructing a projection from semantics to points in high-dimensional spaces. The effect of text sentiment classification mainly depends on extracting compact and informative features from unstructured text through representation learning. Mainstream representation models for text sentiment classification can be divided into two categories based on the knowledge and

information they use: traditional machine learning-based representation models and current popular deep learning-based representation models. The traditional machine learning-based representation models train a sentiment classifier relying on sentiment linguistic knowledge such as bag-of-words and sentiment lexicon, where the sentiment polarity of text is largely determined to be positive if the number of positive words is larger than that of the negative ones. In contrast, the current popular deep learning-based representation models utilize the deep neural network to learn the semantic information contained in text. The performance of deep learning-based representation models is often more superior than that of machine learning-based representation models when the syntactic structure of text is complex. In this paper, we mainly focus on integrating sentiment linguistic knowledge into the deep neural network to enhance the quality of text representation learning for sentiment classification task.

The main idea of the traditional machine learning-based representation models [2] is to employ text classifiers such as the naive Bayes classifier, maximum entropy model, and support vector machines to predict the sentiment polarity of the given texts. The performance of the naive Bayes classifier for text sentiment classification is based on the conditional probability of word-level features belonging to certain sentiment class, where features are manually designed by the bag-of-words representation method [3]. The maximum entropy model is also a statistical machine learning method based on the bag-of-words representation approach and its accuracy of sentiment classification entirely depends on the quality of the hand-crafted corpus so that the process of parameter optimization is computationally intensive and time-consuming [4]. The support vector machines constructs the sentiment feature vector mainly relying on the frequency of occurrence of words in text and trains the decision function to predict the sentiment polarity of sentences [5]. The success of these machine learning algorithms generally relies on improving feature engineering work in terms of the bag-of-words representation learning method and manual sentiment lexicon. For example, considering each sentence as a vector with the following groups of features: n-grams, character n-grams, non-contiguous n-grams, POS tags, cluster n-grams, and lexicon features, support vector machines [6] can beat all strong competitors to be the best performer of traditional machine learning models in the SemEval-2014 task. However, traditional statistical representation learning-based feature engineering work is labor intensive, and it is difficult to breakthrough its performance bottleneck for text sentiment classification tasks in the current field of NLP due to the failure to encode word order and syntactic information. Although the application of sentiment linguistic knowledge in conventional machine learning approaches has reached the upper bound, this phenomenon does not mean that it is not suitable for the current popular deep learning methods. The main goal of our thesis is to explore a way to combine sentiment linguistic knowledge with deep learning methods so as to stimulate the maximum potential of the sentiment linguistic knowledge.

Recently, it has been widely acknowledged that deep learning-based representation models have achieved great success in text sentiment classification tasks [7,8]. This is because deep learning methods are capable of learning text representation from original data without laborious feature engineering work, and capture semantic relations between context words in a scalable manner better than traditional machine learning approaches. As Mikolov et al. [9] proposed, the Word2Vec method which converted each word in the text into a continuous dense vector and distributed the different syntax and semantic features of each word to each dimension in vector space, the deep learning models could apply this method to the word embedding module in order to simplify feature engineering work. Kalchbrenner et al. [10] used a convolutional neural network for modeling sentences to obtain a promising result in text sentiment classification tasks, which demonstrated n-gram features from different positions of a sentence through convolutional operations which could promote the efficiency of predicting sentiment polarity. However, the convolutional neural network completely ignores the sequence information of the text while paying attention to the local features of a sentence. In contrast to the convolutional neural network, a long short-term memory (LSTM) network [11] is good at learning sequential correlation in the text by using an adaptive gating mechanism. Tai et al. [12] verified the

importance of text sequence information learned by standard LSTM to the effect of text sentiment classification and further proposed a Tree-LSTM that incorporates the linguistic syntactic structure of a sentence into this particular structure. Unfortunately, the LSTM structure has lost the ability to learn local features of text, which is an important property of the convolutional neural network. The gated recurrent unit (GRU) [13] network is an improved variant of the LSTM, which has been improved in terms of network structure and performance, but it does not change the congenital defect of LSTM in capturing the text's local features. To avoid ignoring sequence correlation information or context local features when dealing with original unstructured text, it is important to explore an effective combination strategy that takes advantage of the convolutional neural network (CNN) and the GRU network to enrich a sufficient text representation for sentiment classification in our work.

In addition, the attention mechanism is put forward to greatly improve the quality of sentiment representation learning. The seminal NLP work using the attention mechanism is neural machine translation, where different weights are assigned to source words to implicitly learn alignments for translation [14]. The core of the attention mechanism is to imitate human's attention behavior, that a human asked to read a sentence can selectively focus on parts of context words that are important for understanding the sentence. Several recent works have been proposed to design a fusion model of an attention mechanism and a deep learning method to do sentiment representation learning tasks. Zhou et al. [15] proposed a hierarchical attention model which was jointly trained with the LSTM network to capture these key sentiment signals for predicting the sentiment polarity. Zhou et al. [16] made significant progress in extracting deep meaningful sentiment features effectively by combining bidirectional LSTM with an attention mechanism. Du et al. [17] integrated the attention mechanism with a CNN to enhance the quality of extracted local text features. Our work is inspired by the characteristic of attention mechanisms. With the help of an attention mechanism as a bridge, sentiment linguistic knowledge and deep learning methods can be perfectly integrated to enhance the sentiment feature of text.

To alleviate the aforementioned limitations, in this study, we propose a sentiment-feature-enhanced deep neural network (SDNN) for text sentiment classification. First, we propose a novel sentiment attention mechanism that uses a traditional sentiment lexicon as an attention source attending to context words via the attention mechanism. Its goal is to learn more comprehensive and meaningful sentiment-aware sentence representations as input to the deep neural network, establishing an effective relationship between sentiment linguistic knowledge and deep learning methods. Second, we designed a new model based on deep learning, combining GRU and CNN, to further enhance the representation quality of textual structure information. Above all, CNN performs poorly in learning sequential correlation information, while GRU lacks the ability to extract context local features. By using our design, this model can effectively compensate for their own defects and maximize the potential of their respective strengths.

The main contributions of our work are summarized as follows:

1. We leverage traditional sentiment lexicon and a current popular attention mechanism to design a novel sentiment attention mechanism. The sentiment attention mechanism shows strong capability of interactively learning sentiment and context words to highlight the important sentiment features for text sentiment analysis.
2. Both text sequential correlation information generated by recurrent neural network and context local features captured by the convolutional neural network are essential in the text sentiment classification task, so we designed a deep neural network to effectively combine a GRU-based recurrent neural network and a convolutional neural network to enrich textual structure information.
3. Extensive experiments have been conducted on two real-world datasets with a binary-sentiment-label and a multi-sentiment-label to evaluate the effectiveness of the SDNN model for text sentiment classification. The experimental results demonstrate that the proposed SDNN model achieves substantial improvements over the compared methods.

The remainder of this paper is organized as follows. Section 2 presents the SDNN architectures for text sentiment classification in detail, including the feature-enhanced word-embedding module, bidirectional GRU network module, convolutional neural network module, and sentence classifier module. The experiment is introduced in Section 3. Section 4 presents the conclusions and discusses future work.

2. Model

The architecture of the proposed SDNN model is shown in Figure 1, which consists of four key components: the feature-enhanced word-embedding module, bidirectional GRU network module, convolutional neural network module, and the sentence classifier module. In addition, both the bidirectional GRU network module and the convolutional neural network module jointly constitute the deep neural network module. The goal of our model is to predict the sentiment polarity of the given sentence. In the rest of this section, we will elaborate our SDNN model in detail.

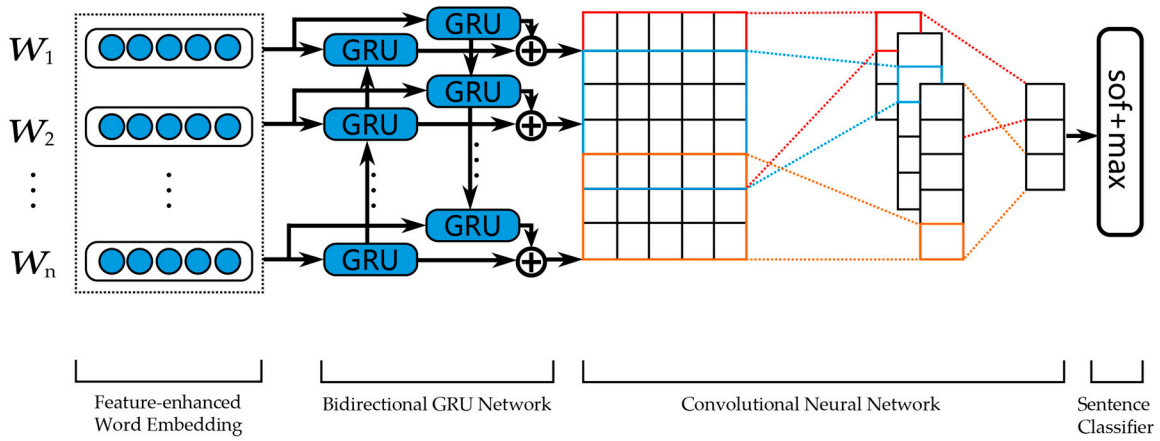

Figure 1. The architecture of the proposed sentiment-feature-enhanced deep neural network (SDNN) model.

2.1. Feature-Enhanced Word Embedding Module

The first step is to pre-process the input sentence and sentiment resource words. In order to transfer each word in the sentence to a real-value vector, we applied the pre-trained GloVe [18] method in the word embedding layer. Let $L \in R^{|V| \times d}$ be the embedding lookup table generated by GloVe, where d is the dimension of word vector and $|V|$ is the vocabulary size. Suppose the input sentence consists of n words and the sentiment resource sequence consists of m words. The input sentence retrieves the word vectors from L and gets a list of vectors $[w_1, w_2, \cdots, w_n]$ where $w_i \in R^d$ is the word vector of the i^{th} word. Similarly, the sentiment resource sequence can retrieve the word vectors from L and form a list of vectors $\left[w_1^s, w_2^s, \cdots, w_m^s\right]$. In this way, we can get the matrix $W^c = [w_1, w_2, \cdots, w_n] \in R^{n \times d}$ for context words and the matrix $W^s = \left[w_1^s, w_2^s, \cdots, w_m^s\right] \in R^{m \times d}$ for sentiment resource words.

After obtaining the general matrix of the word vector, we propose the novel sentiment attention mechanism to help highlight the vital sentiment-resource-relevant context words for generating the sentence representation with enhanced sentiment features. Specifically, we leverage sentiment words in the sentiment lexicon as sentiment resource words and integrate them with the attention mechanism to emphasize more important information related to the sentiment polarity. The sentiment lexicon collects sentiment resource words from both Hu and Liu [19] and Qian et al. [20], including 10,899 sentiment resource words in total. Then, the attention mechanism uses sentiment resource words as an attention source attending to the context words to learn the feature-enhanced word embedding. In the following, we will describe the sentiment attention mechanism in detail.

First, inspired by the fact that the sentiment words can largely guide the sentiment polarity in the context of a sentence, we plan to design the word-level relationship between the sentiment words and the context words. For example, in the sentence "This movie is so wasteful of talent, it is truly disgusting", composed of the sentiment words (i.e., "wasteful" and "disgusting") and the context words (i.e., all words in this sentence except the sentiment words), "wasteful" and "disgusting" play a key role in directing the sentiment polarity of this sentence. Mathematically, we adopt the dot product operation between the context words and the sentiment words to form a correlation matrix. The specific calculation method is as follows:

$$M^s = W^c \circ (W^s)^T \tag{1}$$

where $\circ$ denotes the dot product operation and $M^s \in R^{n \times m}$ represents the relevance matrix between the context words and the sentiment words.

Then, we define the context-word-relevant sentiment word representation matrix X^s generated by the dot product operation between the context words W^c and the correlation matrix M^s:

$$X^s = (W^c)^T \circ M^s \tag{2}$$

where $X^s \in R^{d \times m}$ represents the sentiment word representation matrix related to the context words. Similarly, we can compute the sentiment-word-relevant context word representation matrix X^c by the dot product between the sentiment words W^s and the correlation matrix M^s:

$$X^c = (M^s \circ W^s)^T \tag{3}$$

where $X^c \in R^{d \times n}$ represents the context word representation matrix related to the sentiment words. The illustration of sentiment-context word correlation is shown in Figure 2.

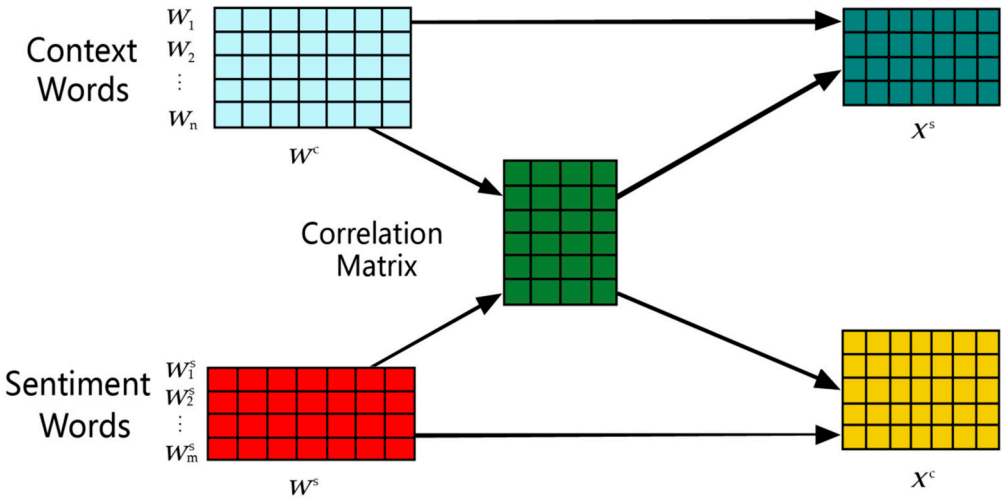

Figure 2. Sentiment-context word correlation.

After obtaining the sentiment-context word correlation, we adopt the attention mechanism to highlight the information contributing to predicting the sentiment polarity of the input sentence. As described in this section, we consider the sentiment influence on the context from the sentiment words, which can provide more clues to pay attention to the related sentiment features. Meanwhile, we can handle some complex situations with changing sentiment by using the attention mechanism. For example, in the sentence "It is actually my favorite kind of film, but as an adaptation, it fails from every angle", a real-world man will focus on the word "favorite" after reading the first clause of this sentence. Because "favorite" is the word that largely represents the sentiment polarity of the current

sentence. Until the last word in this sentence is read, the man's attention will turn to the word "fail", which determines the sentiment polarity of the whole sentence. By using the attention mechanism that can simulate the real-world man's attention, "fails" in the context of "it fails from every angle" will be assigned more "attention" than "favorite" in the context of "my favorite kind of film". Formally, the attention score function β is defined as follows:

$$t_s = \frac{\sum_{i=1}^{m} X_i^s}{m} \tag{4}$$

$$\beta\left(\left[X_i^c; t_s\right]\right) = u_s^T \tan \mathrm{h}\left(W_s\left[X_i^c; t_s\right]\right) \tag{5}$$

where t_s denotes the overall representation of sentiment words, u_s^T and W_s are learnable parameters. With the attention score function β that calculates the importance of the ith word X_i^c in the context, the attention mechanism generates the attention vector α_i by:

$$\alpha_i = \frac{\exp\left(\beta\left(\left[X_i^c; t_s\right]\right)\right)}{\sum_{i=1}^{n} \exp\left(\beta\left(\left[X_i^c; t_s\right]\right)\right)} \tag{6}$$

where α_i represents the importance of the ith word in the sentence. Finally, we can attend the attention vector α_i to the context words:

$$x_i = \alpha_i X_i^c \tag{7}$$

where x_i represents the ith word of the original input sentence. The final output of the feature-enhanced word-embedding layer is $X = [x_1, x_2, \cdots, x_n]$. Although the attention mechanism can highlight the sentiment features in the sentence, the model cannot fully capture the interactive information of the textual structure. In order to enhance the final representation of the sentence, we pass the sentence representation generated by the feature-enhanced word-embedding module to the deep neural network module.

2.2. Deep Neural Network Module

The deep neural network module is composed of two parts: Bi-GRU and CNN. As we all know, text is structured and organized. With the purpose of avoiding the destruction of the sequence structure of text, our model passes the output of feature-enhanced word-embedding layer to the Bi-GRU layer and then to the CNN layer, and obtain the final representation of the input sentence. In the following, we will detail the two parts in the order of the data flow.

As shown in Figure 1, the Bi-GRU layer contains two sub-layers for the forward and backward sequences, respectively, which is beneficial to have access to the future context as well as the past context. Since the GRU unit is a variant of LSTM, we first briefly review the LSTM for the sequence modeling task. The main idea of the LSTM is to overcome the problem of gradient vanishing and expansion in the recurrent neural network by introducing an adaptive gating mechanism that controls the data flow to and from their memory cell units. Taking the sequence vector $X = [x_1, x_2, \cdots, x_n]$ from the output of feature-enhanced word-embedding layer as an example, LSTM processes the data word-by-word corresponding to the time-step from the past to the future. At time step t, the current hidden state h_t and the current memory-cell state c_t are calculated as follows:

$$i_t = sigmoid\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{8}$$

$$f_t = sigmoid\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{9}$$

$$o_t = sigmoid\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{10}$$

$$\widetilde{c_t} = tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right) \tag{11}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c_t} \tag{12}$$

$$h_t = o_t \odot \tan \mathrm{h}(c_t) \tag{13}$$

where $\cdot$ denotes matrix multiplication and $\odot$ stands for element-wise multiplication. i_t refers to the input gate which controls the input of new information to the memory cell, f_t indicates the forget gate, which controls how long certain values are held in the memory cell, and o_t represents the output gate which controls how much the values stored in the memory cell affect the output activation of the block. W_i, W_f, and W_o are the weight matrixes for these three gates, respectively, while b_i, b_f, b_o are the bias vectors for these gates.

As for GRU, it is regarded as an optimization of LSTM, and it can not only merge the input gate i_t and the forget gate f_t of LSTM into the reset gate r_t, but also optimize the update mode of the hidden state h_t. In addition, the output gate o_t corresponds to the update gate z_t. Throughout this design, the GRU can maintain the advantage of the LSTM while having a simpler structure, fewer parameters, and better convergence than the LSTM [13]. As shown in Figure 3, at each time-step t, the GRU transition functions are defined as follows:

$$r_t = sigmoid\ (W_r \cdot [h_{t-1}, x_t]) \tag{14}$$

$$z_t = sigmoid\ (W_z \cdot [h_{t-1}, x_t]) \tag{15}$$

$$\widetilde{h_t} = tanh\left(W_{\widetilde{h}} \cdot [r_t \odot h_{t-1}, x_t]\right) \tag{16}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h_t} \tag{17}$$

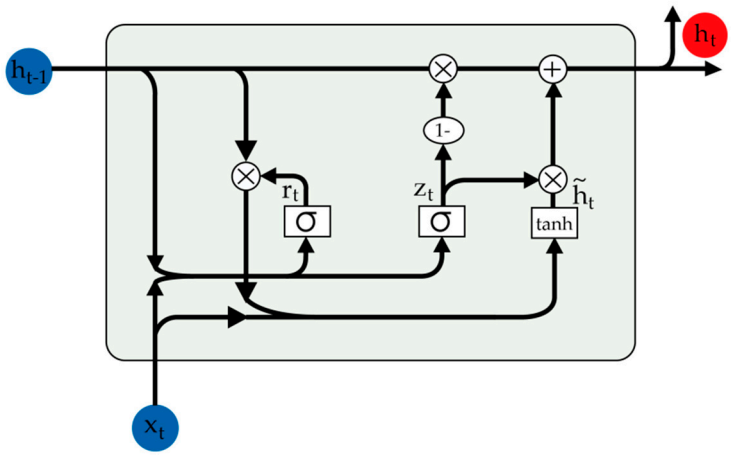

Figure 3. The architecture of the gated recurrent unit (GRU) cell used in the SDNN model.

In the Bi-GRU layer, the forward GRU layer processes the sentence word-by-word in the order of the input sequence to obtain the hidden state $\overrightarrow{h_t}$ at each time-step t. The backward GRU layer does the same thing, except that its input sequence is reversed. The final hidden state h_t at time step t is updated as:

$$\overrightarrow{h_t} = \overrightarrow{GRU}\ (x_t,\ \overrightarrow{h_{t-1}}) \tag{18}$$

$$\overleftarrow{h_t} = \overleftarrow{GRU}\ (x_t,\ \overleftarrow{h_{t-1}}) \tag{19}$$

$$h_t = [\overrightarrow{h_t} \oplus \overleftarrow{h_t}] \tag{20}$$

where $\oplus$ denotes element-wise sum between the forward hidden state $\overrightarrow{h_t}$ and the backward hidden state $\overleftarrow{h_t}$. The output of the Bi-GRU layer is $H = [h_1, h_2, \cdots, h_n]$, where n is the length of the input sentence.

Although the sentence representation obtained by the Bi-GRU layer maintains the sequence information of the sentence, it is not flexible enough to predict the sentiment polarity of the input

sentence. To alleviate this problem, the SDNN model feeds the output of the Bi-GRU layer into the CNN layer. This is because the CNN has the ability of recognizing the local features inside a multi-dimensional field. Specifically, the CNN layer consists of a one-dimension convolutional layer and a max-pooling layer. First, we can treat $H = [h_1, h_2, \cdots, h_n] \in R^{n \times d}$, where h_i represents the i^{th} word in the sentence with d dimension, as an "image" like in Figure 1, so the one-dimension convolutional layer can slide along the word dimension to convolve the matrix H with multiple kernels of different widths. The convolutional operation can be represented as follows:

$$c_i = f\left(H_{i:i+K} \circ W_c + b_c\right) \tag{21}$$

where $\circ$ denotes the dot product operation, K represents the width of convolutional kernel, f is a non-linear function such as ReLU, W_c is the convolutional matrix, and b_c is the bias term. Each kernel corresponds to a linguistic feature detector which extracts a specific pattern of n-gram at various granularities [10]. The convolutional kernel is applied to each possible region of the matrix H to produce a feature map $C = [c_1, c_2, \cdots, c_{n_K}]$ for the same width of convolutional kernel, where n_K is the number of convolutional kernels. Then for each c_i in C, the max pooling layer extracts the maximum value from the generated feature map:

$$p_i = down\left(c_i\right) \tag{22}$$

where $down\left(\cdot\right)$ represents the max pooling function. Through this way, the pooling layer can extract the local dependency within different regions to keep the most salient information, and it results in a fixed-size vector whose size is equal to n_K. Finally, the output of the max pooling layer with different widths is concatenated to form the final sentence representation $S^* = \left[p_1, p_2, \cdots, p_{n_K \times n_{wid}}\right]$, where n_{wid} denotes the number of different width.

In the deep neural network module, the sequence information generated by Bi-GRU and the local feature captured by CNN are integrated with an effective strategy of combining Bi-GRU and CNN, which can help the sentence classifier to predict the sentiment polarity.

2.3. Sentence Classifier Module

For text sentiment classification, the final sentence representation S^* of the input text S is fed into a softmax layer to predict the probability distribution of sentence sentiment label over C (number of sentiment category labels), and the sentiment label with the highest probability is selected as the final sentiment category to which the sentence belongs. The function is shown as follows:

$$\widetilde{y} = \frac{exp\left(W_o^T\, s^* + b_o\right)}{\sum_{i=1}^{C} exp\left(W_o^T\, s^* + b_o\right)} \tag{23}$$

$$y_{pre} = argmax\, \widetilde{y} \tag{24}$$

where $\widetilde{y}$ is the predicted sentiment distribution of the sentence, y_{pre} is the selected sentiment label, W_o^T and b_o are the parameters to be learned.

In order to train the SDNN model, we adopt the categorical cross-entropy loss function as the reasonable training objective which is minimized in the training process:

$$J(\theta) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_i^j \log \widetilde{y}_i^j + \lambda_r\left(\sum_{\theta \in \Theta} \theta^2\right) \tag{25}$$

where N is the training set, y is the one-hot distribution of the ground truth, λ_r is the coefficient for the L_2 regularization, and Θ is the parameter set which includes all the parameters that need to be

optimized during the training process. All the parameters are updated by the stochastic gradient descent strategy, which is defined as:

$$\Theta = \Theta - \lambda_l \frac{\partial J(\theta)}{\partial \Theta} \tag{26}$$

where λ_l is the learning rate. The specific hyper-parameter settings are described in Section 3.2.

3. Experiments

3.1. Datasets

We conduct experiments on two publicly available datasets. The movie review (MR) dataset is a movie review dataset with one sentence per review collected by Pang and Lee [21], which aims to detect positive or negative reviews. The MR dataset has 5331 negative samples and 5331 positive samples. The Stanford Sentiment Treebank (SST) dataset is an extension of the MR by Socher et al. [22], which is manually separated from the train, development, test sets, and contains fine-grained sentiment labels (very positive, positive, neural, negative, very negative). Similar to the sample distribution of the MR dataset, the number of instances of each class in the SST dataset is approximately equal, which can effectively avoid the reduction of model generalization ability caused by the uneven distribution of dataset samples in the process of model training. In particular, since the MR dataset lacks a development set, we randomly sampled 10% of the training data as the development set. The detailed dataset statistics are shown in Table 1.

Table 1. The summary statistics of the two datasets. c: number of target classes, l: average sentence length, m: maximum sentence length, train/dev/test: train/development/test set size, |V|: vocabulary size, $|V_{pre}|$: number of words present in the set of pre-trained word embeddings, CV: 10-fold cross validation.

Dataset	c	l	m	train	dev	test	|V|	$\|V_{pre}\|$
Movie review (MR)	2	21	59	10,662	-	CV	20,191	16,746
Stanford Sentiment Treebank (SST)	5	18	51	8544	1101	2210	17,836	12,745

3.2. Implementation Details

In order to improve the quality of the dataset, we pre-processed the text data by removing stopwords (e.g., "in", "of", "from") and punctuation. Then, all word embeddings from the text data were initialized by 200-dimensional GloVe vectors pre-trained by Pennington et al. [18]. For the out-of-vocabulary words, we randomly sampled their embeddings from a uniform distribution U (−0.1, 0.1). Some works adopted the fine-tuned training strategy for word vectors to improve the performance for text sentiment classification tasks [23]. In contrast, with the purpose of better reflecting the generalization ability of the model, we preferred to use the general embeddings for all datasets. What is more, we treated all the context words as sentiment resource words to implement the self-attention mechanism as if there was no sentiment resource word in the sentence.

For the deep neural network, the hidden states of the GRU unit in each layer were set to 200. In the convolution layer, we employed 1D convolutional filter windows of 3, 4, and 5 with 100 feature maps each, and a 1D pooling size of 4.

During the training process, we optimized the proposed model with the AdaDelta algorithm [24] by following the learning rate of 10^{-2} and the mini-batch size of 32. To alleviate the overfitting problem, we employed the dropout strategy [25], with a dropout rate of 0.5 for the Bi-GRU layer, 0.2 for the penultimate layer, and 10^{-5} for the coefficient λ_r of L_2 regularization. To evaluate the performance of the sentiment classification task, we used Accuracy and F1 as the metrics.

3.3. Baseline Methods

In order to comprehensively evaluate the performance of the SDNN, we set several strong baseline methods for sentiment classification, including traditional machine learning models, traditional deep learning models, deep learning models relying on specified parsing structure, and models extracting important information by attention mechanisms.

SVM/Feature-SVM: SVM [6] is a classic machine learning method to solve sentiment classification tasks, which won first place in the SemEval-2014 Task 4 by using the following groups of features: n-grams, character n-grams, non-contiguous n-grams, POS tags, cluster n-grams, and lexicon features. In addition, we incorporated sentiment resource words into the implementation of the SVM model (denoted as Feature-SVM).

LSTM/Bi-LSTM: LSTM/Bi-LSTM (Bidirectional Long Short-Term Memory) [12] has the capability to acquire dependencies between words in a sentence by incorporating the long short-term memory unit and the bidirectional variant into neural networks, which makes it effective in solving the problem of the long dependence of text.

CNN: A convolutional neural network [26] is a very strong baseline for text sentiment classification tasks. It can sufficiently capture the local feature information of the text to generate task-specific sentence representation.

BLSTM-C: BLSTM-C (Bi-LSTM combined with the generalized CNN) [27] combines CNN with Bi-LSTM networks in order to fuse the local information and sequence information of text to form feature-enhanced sentence representation for predicting the sentiment polarity of a sentence.

Tree-LSTM: Tree-LSTM (Tree-structured long short-term memory) [12] introduced memory information into tree-structured neural networks, relying on predefined paring structures, which helps to capture the semantic relatedness.

LR-Bi-LSTM: LR-Bi-LSTM (Linguistically Regularized-based Bidirectional Long Short-Term Memory) [20] makes use of linguistic roles with neural networks by utilizing linguistic regularization on intermediate outputs with KL divergence.

Self-Attention: Self-attention [28] can learn structured sentence embedding with a special regularization term.

3.4. Experimental Results

3.4.1. Overall Performance

Experimental results listed in Table 2 show that our proposed model (SDNN) performs competitively on MR and SST datasets. We can draw the following observations from Table 2.

First, we can see that the Feature-SVM outperformed the SVM on two datasets by roughly 0.9% on accuracy and 0.8% on F1. It shows that the SVM equipped with sentiment linguistic knowledge can better learn the representation of text. It also means that the sentiment linguistic knowledge is important to text sentiment classification tasks. Obviously, compared with Feature-SVM on two datasets, our SDNN model improved the accuracy by roughly 8.0% and F1 by roughly 7.3%. The main reason is that traditional machine learning-based methods pay more attention to word frequency features while ignore the context structure information.

Second, the conventional deep learning models (i.e., LSTM, Bi-LSTM, and CNN) outperformed Feature-SVM by a large margin on the MR and SST datasets. Although the performance of LSTM on the MR dataset was similar to that of the Feature-SVM, the accuracy and F1 of the LSTM on the SST dataset were improved by 4.9% and 4.3%, respectively. The LSTM was the worst performer of the conventional deep learning models. Undoubtedly, compared with the traditional deep learning models (i.e., LSTM, Bi-LSTM, and CNN), BLSTM-C had better results by effectively combining the CNN and LSTM networks, because it can simultaneously learn the sequence structure information of the text and capture the local features of the text, which helps to better understand the text structure information.

Table 2. Evaluation results (%) for the MR and SST datasets. The best result for each dataset is in bold. Results marked with # were retrieved from the references (i.e., [20,22,26,27]), while those unmarked with # were obtained either by our own implementation or with the same codes shared by the original authors.

Models	MR		SST	
	Accuracy	F1	Accuracy	F1
SVM	76.4	78.8	40.7 #	42.4
Feature-SVM	77.3	79.4	41.5	43.3
LSTM	77.4 #	79.6	46.4 #	47.6
Bi-LSTM	79.3 #	81.0	49.1 #	50.5
CNN	81.5 #	82.7	48.0 #	49.3
BLSTM-C	82.4	83.8	50.2 #	52.0
Tree-LSTM	80.7 #	82.1	50.1 #	51.8
LR-Bi-LSTM	82.1 #	83.6	50.6 #	52.3
Self-Attention	81.7	82.9	48.9	50.1
SDNN	**83.7**	**84.9**	**51.2**	**52.9**
SDNN w/o sentiment attention	82.5	84.1	50.0	51.5

Further, the deep learning models with parsing structures (i.e., Tree-LSTM and LR-Bi-LSTM) outperformed the LSTM model on the two datasets by roughly 4.0% on accuracy and 3.8% on F1. These examples demonstrate that integrating external knowledge into deep neural networks can have a better understanding of the input text for sentiment analysis. As we expected, the SDNN achieved the best performance among all the strong competitors on the MR and SST datasets. Compared with the BLSTM-C, which was the best performer of all of the baseline methods on the same datasets, our SDNN model upgraded the results by about 1.2% on accuracy and 1.0% on F1. The results confirm our main idea that integrating sentiment linguistic knowledge into the deep neural network can enhance the quality of text representation learning for sentiment classification.

In order to analyze the effectiveness of the sentiment attention mechanism of SDNN, we also designed the ablation test in terms of discarding the sentiment attention mechanism (denoted as SDNN w/o sentiment attention). It can be clearly seen from the experimental results that the accuracy and F1 of the SDNN decreased sharply without regard to the sentiment attention mechanism. This proves our intuition that the proposed sentiment attention mechanism plays a crucial role in SDNN for text sentiment classification.

3.4.2. Effects of Different Combinations of Bi-GRU and CNN

To investigate the effectiveness of different combinations of Bi-GRU and CNN, we designed a series of models to compare the different effects of different combinations.

Bi-GRU+CNN: The Bi-GRU+CNN model proposed in the paper.

CNN+Bi-GRU: Based on Bi-GRU+CNN, the output of the word-embedding layer passes through the deep neural network in the order of the first CNN and the second Bi-GRU.

CNN-Bi-GRU: On the basis of Bi-GRU+CNN, the output of the word-embedding layer inputs into the Bi-GRU and CNN, respectively, and then the outputs of the networks are concatenated to form a representation of the sentence.

The results are shown in Table 3. We can see that the performance of CNN-Bi-GRU is mediocre relative to the other two models, but the 6.7% difference between Bi-GRU+CNN and CNN+Bi-GRU is not coincidental. The main reason is that the initial convolutional layer of CNN+Bi-GRU destroyed the sequence structure of the text so that the Bi-GRU layer behind it was just like a fully connected layer, which fails to harness the full capabilities of the Bi-GRU layer. The CNN+Bi-GRU was even worse than a regular LSTM by 0.4%. On the other hand, Bi-GRU+CNN achieved the best performance among the

three variant models. This was because the initial Bi-GRU layer can encode every token in the input text into a vector that contains not only the information of the original token but also the information of the previous token. In this way, the order of each token in the generated text representation is the same as the order in the original text. Afterwards, the CNN layer will find local patterns by using the generated representation, which can further improve the accuracy. This proves that the combination of Bi-GRU and CNN is very sensitive to the improvement of deep neural network performance.

Table 3. Test accuracies (%) of different combinations of Bi-GRU and CNN.

Models	MR	SST
Bi-GRU+CNN	83.7	51.2
CNN+Bi-GRU	77.0	46.1
CNN-Bi-GRU	81.9	48.5

3.4.3. Effects of the Dimension of Sentiment Resource Words

Since the sentiment attention mechanism was added in the word-embedding layer, in order to verify the influence of different dimensions of sentiment resource words on the classification accuracy, we used SDNN to analyze the accuracy under different dimensions on the MR dataset. The experimental results are shown in Figure 4.

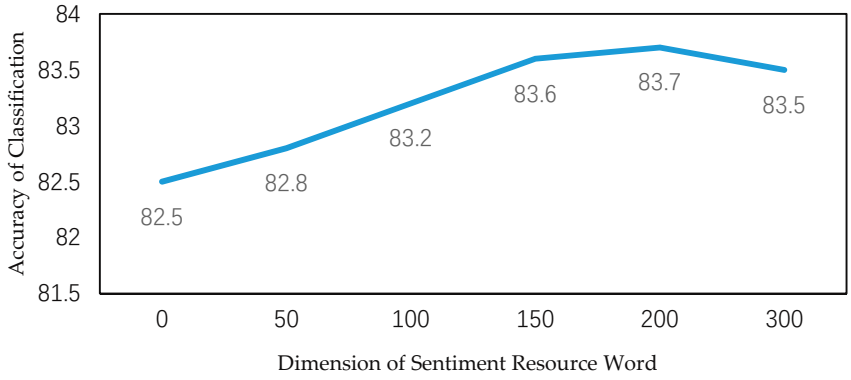

Figure 4. Classification of different dimensions of sentiment resource words on the MR dataset. When the dimension of the word vector is equal to 0, it means that the sentiment attention mechanism was not used in this model.

From Figure 4, we can see that when the dimension of the word vector is less than 200, the accuracy increases rapidly with the increase of the dimension. One main reason is that the model can adjust more component parameters of the sentiment resource word vector to learn the sentiment information of the sentence along with the increase of the dimension of the word vector. However, when the dimension of the word vector exceeds 200, the classification accuracy will fluctuate instead. This phenomenon is related to the meaning of the dimension of the word vector. In a word vector, each dimension represents a deep semantic feature, which is obtained by pre-training a specific corpus. Since the parameters of a word vector remain unchanged during the training process, only the parameters of the attention mechanism associated with it are optimized. In other words, when the dimension of the word vector exceeds a certain threshold, the word vector is burdened with many irrelevant parameters of vector dimension. For the optimization of the sentiment attention mechanism, these parameters are redundant, which will affect the parameter adjustment of the sentiment attention mechanism so as to affect the learning quality of sentiment linguistic knowledge. Thus, we used 200 as the dimension of the sentiment resource word vector in the experiment.

3.5. Visualization of Attention Mechanism

We picked a comment sentence with changing sentiment features as a case study and visualized the attention results. To make the visualized results comprehensible, we removed the deep neural network module to make the sentiment attention mechanism directly work on the word embedding, thus we can check whether the effect of the attention mechanism conforms with human's attention performance. The visualization results are shown in Figure 5.

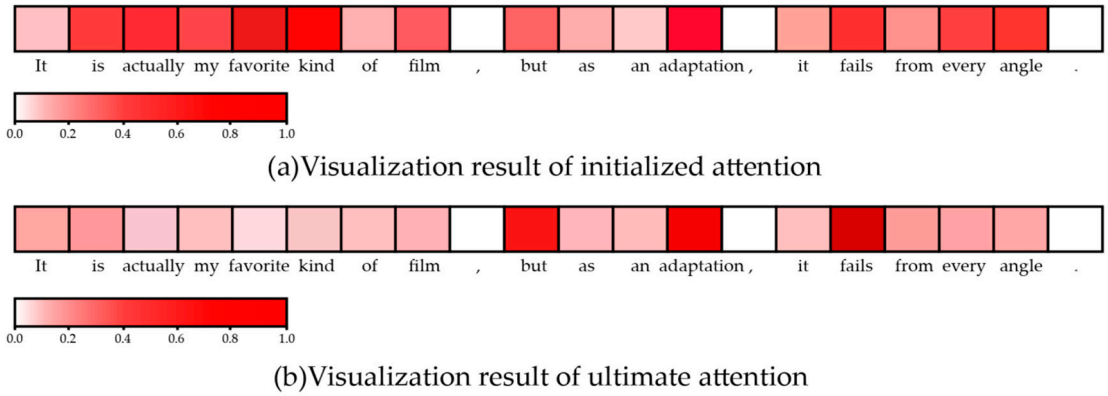

Figure 5. Comparison of initialized attention and ultimate attention. The attention score calculated by Equation (6) is used for the color-coding. The color depth indicates the importance degree of a word to the sentiment polarity of a sentence.

In Figure 5a, we can find that the distribution of attention weights is relatively uniform throughout the sentence when the attention mechanism is initialized. This is because the parameters of the attention mechanism are uniformly distributed during the initialization stage, which makes the attention score of each word similar. This phenomenon is similar to the performance of human's attention, when a human is asked to read a sentence, he will always give an overview of all the words. As the training process progresses, the attention mechanism gradually focuses on words that contribute a lot to the overall sentiment polarity of the sentence. This is related to the supervised learning process, in which the attention mechanism interactively learns the relationship between the context and the sentiment words to optimize the performance of text sentiment classification. In the same way, after reading a sentence, humans only pay attention to several key parts of this sentence, which helps to understand the general meaning of this sentence. As shown in Figure 5b, the ultimate result of the attention mechanism mainly highlights three words (i.e., "fails", "but", and "adaptation"), which are in line with the sentiment polarity of this sentence. Finally, with the purpose of improving the quality of word embedding, the sentiment attention mechanism is combined with the conventional word embedding, which serves as the input layer of the deep neural network to extract text structure information. From the perspective of the overall training process of the model, the sentiment attention mechanism will highlight distinguishable words contributing to the orientation of sentiment polarity, so that the model can avoid blindly learning all the context words in the next step.

4. Conclusions and Future Work

In this paper, we proposed the sentiment-feature-enhanced deep neural network for text sentiment classification tasks by integrating sentiment linguistic knowledge into the deep neural network via a sentiment attention mechanism. We implemented the novel sentiment attention mechanism by combining a traditional sentiment lexicon with an attention mechanism to learn sentiment-feature-enhanced word representation, bridging the knowledge gap between conventional sentiment linguistic knowledge and the deep learning method. In addition, we also designed a deep neural network to effectively combine local features within the sentence captured by the convolutional neural network and the sequence information across a sentence generated by a bidirectional GRU

network, which can further improve the quality of text representation for achieving greater promotion of text sentiment classification tasks. Extensive experiments were conducted on real-world datasets with a binary-sentiment-label and a multi-sentiment-label. The experimental results showed that the proposed SDNN significantly outperformed the state-of-the-art methods for text sentiment classification tasks.

In the future, we will try to use this model to analyze the actual emotions of specific users, such as sadness, happiness or depression, which can lay a good foundation for providing better specific users' experience in terms of the marketing service industry. Furthermore, it can be seen from experimental results that the proposed model still has much room for improvement on the dataset with fine-grained sentiment labels. Therefore, the reinforcement learning method to build structured text representation without explicit linguistic structure annotations will be tried in our experiment to see if it will achieve better performance.

Author Contributions: W.L. (Wenkuan Li), P.L., and Q.Z. conceived and designed the study. W.L. (Wenkuan Li), Q.Z., and W.L. (Wenfeng Liu) performed the experiments. P.L. provided the data. W.L. (Wenkuan Li) and Q.Z. analyzed the data. W.L. (Wenkuan Li) and W.L. (Wenfeng Liu) proofed the algorithm. W.L. (Wenkuan Li) wrote the paper. All authors read and approved the paper.

Acknowledgments: This work was supported by the National Natural Science Foundation of China (Grant No. 61373148), the Science Foundation of the Ministry of Education of China (No. 14YJC860042), and the Shandong Provincial Social Science Planning Project (No. 18CXWJ01/18BJYJ04/17CHLJ33/17CHLJ30/17CHLJ18).

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

References

1. Bing, L. Sentiment Analysis and Opinion Mining. In *Encyclopedia of Machine Learning and Data Mining*; Springer: Boston, MA, USA, 2012; Volume 30, p. 167.
2. Wang, S.; Manning, C.D. Baselines and bigrams: Simple, good sentiment and topic classification. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Jeju Island, Korea, 8–14 July 2012; Volume 2, pp. 90–94.
3. Jiang, Q.W.; Wang, W.; Han, X. Deep feature weighting in naive Bayes for Chinese text classification. In Proceedings of the 2016 Fourth International Conference on Cloud Computing and Intelligence Systems, Beijing, China, 17–19 August 2016; pp. 160–164.
4. Yin, C.; Xi, J. Maximum entropy model for mobile text classification in cloud computing using improved information gain algorithm. *Multimed. Tools Appl.* **2016**, *76*, 1–17. [CrossRef]
5. Joachims, T. Transductive inference for text classification using support vector machines. In Proceedings of the Sixteenth International Conference on Machine Learning, Bled, Slovenia, 27–30 June 1999; pp. 200–209.
6. Kiritchenko, S.; Zhu, X.; Cherry, C.; Saif, M. Detecting aspects and sentiment in customer reviews. In Proceedings of the 8th International Workshop on Semantic Evaluation, Dublin, Ireland, 23–24 August 2014; pp. 437–442.
7. Gatt, A.; Krahmer, E. Survey of the state of the art in natural language generation: core tasks, applications and evaluation. *Vestn. Oftalmol.* **2017**, *45*, 1–16. [CrossRef]
8. Young, T.; Hazarika, D.; Poria, S. Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Comput. Intell. Mag.* **2017**, *13*, 55–75. [CrossRef]
9. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; Volume 2, pp. 3111–3119.
10. Nal, K.; Edward, G.; Phil, B. A convolutional neural network for modelling sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 23–25 June 2014; pp. 655–665.
11. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]

12. Tai, K.S.; Socher, R.; Manning, C.D. Improved semantic representations from tree-structured long short-term memory networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 30–31 July 2015; pp. 1556–1566.
13. Cho, K.; Merrienboer, B.V.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Proceedings of the Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1724–1735.
14. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
15. Zhou, X.J.; Wan, X.J.; Xiao, J.G. Attention-based lstm network for cross-lingual sentiment classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 247–256.
16. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; p. 207.
17. Du, J.; Gui, L.; Xu, R.; He, Y. A convolutional attention model for text classification. In Proceedings of the National CCF Conference on Natural Language Processing and Chinese Computing, Dalian, China, 8–12 November 2017; pp. 183–195.
18. Pennington, J.; Socher, R.; Manning, C. Glove: Global vectors for word representation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
19. Hu, M.Q.; Liu, B. Mining and summarizing customer reviews. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 22–25 August 2004; pp. 168–177.
20. Qian, Q.; Huang, M.; Lei, J. Linguistically Regularized LSTMs for Sentiment Classification. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1679–1689.
21. Pang, B.; Lee, L. A Sentimental Education: Sentiment Analysis Using Subjectivity, Summarization Based on Minimum Cuts. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain, 21–26 July 2004; pp. 271–278.
22. Socher, R.; Perelygin, A.; Wu, J.Y.; 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, Seattle, WA, USA, 18–21 October 2013; Volume 1631, p. 1642.
23. Zhang, R.; Lee, H.; Radev, D. Dependency Sensitive Convolutional Neural Networks for Modeling Sentences and Documents. In Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational, San Diego, CA, USA, 12–17 June 2016; pp. 1512–1521.
24. Andrychowicz, M.; Denil, M.; Gomez, S.; Hoffman, M.W.; Pfau, D.; Schaul, T.; Shillingford, B.; Freitas, N.D. Learning to learn by gradient descent by gradient descent. *arXiv* **2016**, arXiv:1606.04474.
25. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
26. Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
27. Li, Y.; Wang, X.; Xu, P. Chinese Text Classification Model Based on Deep Learning. *Future Internet* **2018**, *10*, 113. [CrossRef]
28. Gui, L.; Zhou, Y.; Xu, R.; He, Y.; Lu, Q. Learning representations from heterogeneous network for sentiment classification of product re-views. *Knowl. Based Syst.* **2017**, *124*, 34–45. [CrossRef]