electronics

MDPI

Article

Post Text Processing of Chinese Speech Recognition Based on Bidirectional LSTM Networks and CRF

Li Yang [1], Ying Li [1], Jin Wang [1,2,*] and Zhuo Tang [3,4]

1 Hunan Provincial Key Laboratory of Intelligent Processing of Big Data on Transportation, School of Computer & Communication Engineering, Changsha University of Science & Technology, Changsha 410004, China; yanglixt@126.com (L.Y.); liying@stu.csust.edu.cn (Y.L.)
2 School of Information Science and Engineering, Fujian University of Technology, Fuzhou 350118, China
3 College of Information Science and Engineering, Hunan University, Changsha 410082, China; ztang@hnu.edu.cn
4 National Supercomputing Center in Changsha, Hunan University, Changsha 410082, China
* Correspondence: jinwang@csust.edu.cn; Tel.: +86-180-1484-9250

Received: 28 September 2019; Accepted: 28 October 2019; Published: 31 October 2019

Abstract: With the rapid development of Internet of Things Technology, speech recognition has been applied more and more widely. Chinese Speech Recognition is a complex process. In the process of speech-to-text conversion, due to the influence of dialect, environmental noise, and context, the accuracy of speech-to-text in multi-round dialogues and specific contexts is still not high. After the general speech recognition technology, the text after speech recognition can be detected and corrected in the specific context, which is helpful to improve the robustness of text comprehension and is a beneficial supplement to the speech recognition technology. In this paper, a text processing model after Chinese Speech Recognition is proposed, which combines a bidirectional long short-term memory (LSTM) network with a conditional random field (CRF) model. The task is divided into two stages: text error detection and text error correction. In this paper, a bidirectional long short-term memory (Bi-LSTM) network and conditional random field are used in two stages of text error detection and text error correction respectively. Through verification and system test on the SIGHAN 2013 Chinese Spelling Check (CSC) dataset, the experimental results show that the model can effectively improve the accuracy of text after speech recognition.

Keywords: error detection; error correction; LSTM; CRF; Chinese speech recognition

1. Introduction

With the development of wireless networks, Internet of Things applications emerge in endlessly [1–6]. Among them, speech recognition technology is one of the most widely used. However, the accuracy of speech recognition technology still needs to be improved. Due to the complexity of Chinese, this problem is more prominent. Fortunately, with the development of Internet technology and artificial intelligence, speech recognition technology has received more and more attention. However, due to the accuracy of speech recognition technology and the grammatical habits of the language itself, the results of speech conversion often have some errors. These errors not only affect the subsequent speech recognition, but also increase the difficulty for the recipient to understand the semantics of the speech text. Therefore, it is an important research direction to improve the performance of speech recognition technology to use natural language processing technology for text processing after speech recognition.

At present, the common text processing methods after speech recognition are mainly divided into three categories: dictionary-based, traditional machine learning, and deep learning method. The dictionary-based method [7] aims to build a huge dictionary, find the corresponding words in the

dictionary through string matching, and then check and correct the possible errors [8]. Chiu et al. [9] proposed a Chinese text error correction model based on a combination of rules and statistical methods. They use a rule-based detection model to identify errors in Chinese text and then use statistical machine translation models to provide the most appropriate corrections for erroneous text. Yeh et al. [10] proposed a Chinese spelling check method based on a string matching algorithm and Chinese model. First, the probability of the sentence constructed by the author was detected using the n-gram, and then the string matching algorithm Knuth–Morris–Pratt (KMP) was used to detect and correct the error. Hasan et al. [11] used a character-based statistical machine translation model to correct user search query errors in the e-commerce world. Instead of using the common method of segmenting Chinese sentences first, they used character-by-character processing of Chinese sentences to obtain better performance than the former. Hsieh et al. [12] proposed a word lattice decoding model to solve the Chinese spell check problem. They prevent word segmentation errors and misspelling errors by including all confusing characters in the word grid to improve the accuracy of rule matching. Siklósi et al. [13] proposed a method for automatically correcting spelling errors in Hungarian clinical records. They modeled the spelling correction problem as a translation task, using the error text as the source language, the target text as the corrected text, using the statistical machine translation model to perform the error correction task, and modeling the lexical context using a 3-gram-based language model. However, the accuracy of error detection and error correction based on dictionary is very high, but with a poor robustness. The dictionary requires a lot of manual maintenance. With the rapid development of Internet technology, network vocabulary and new vocabulary emerges endlessly, which brings great difficulties to the maintenance of dictionaries.

With the weakness of dictionary-based methods, many researches adopted traditional machine learning methods. They mainly extract the correlations and features between words from a large number of corpora, train the language model, and finally obtain the optimal correction of words through the language model [14]. Han et al. [15] proposed a Chinese spelling check model based on maximum entropy (ME). The ME was trained as a Chinese spell check model by importing a large original corpus. Zhao et al. [16] used a hybrid model based on a graph-based generic error model and two independently trained specific error models to detect Chinese spelling errors. In the graph-based generic error model, a directed acyclic graph was generated for each sentence, and then a single source shortest path algorithm was used to detect and correct common spelling errors in the sentence. Chen et al. [17] proposed a new Chinese Spelling Check (CSC) probability framework based on the advantages of alternative models and language models. An unsupervised way was used to integrate the topic language model into the CSC system so that semantic information under a very large span could be obtained in a string. In addition, the CSC framework and web resources are integrated to further improve the overall performance of the framework. Liu et al. [18] proposed a Chinese spelling check framework that automatically detects and corrects Chinese spelling errors. Machine translation models and language models were used as part of the framework to generate correction candidates, and then the candidate correction candidates generated by statistical machine translation (SMT) and language model (LM) were sorted by support vector machine (SVM) classifiers. Yeh et al. [19] proposed a method for detecting and correcting Chinese spelling errors using an inverted index table with a reordering mechanism. To reduce the search space and computational complexity of the framework, the context-dependent confidence-based pruning method was applied. With the class-like language model and the maximum entropy correction model, the correlation model between the original input and the expected input was obtained. However, the functions used by the traditional machine learning model in the modeling process are relatively simple, and there are certain limitations on the feature extraction of complex problems.

In recent years, many researches have applied deep learning methods to natural language processing. The deep learning method mainly extracts the features of the large-scale corpus through the neural network model, and then uses the trained model to judge the optimal corrective words [20]. The traditional machine learning model extracts sample features usually in the form of numbers,

so only shallower data features can be extracted, while the deep neural network can better fit the data by combining the underlying features into a more abstract high-level feature representation [21,22]. Therefore, deep neural networks usually have more efficient performance than traditional machine learning models [23]. Wang et al. [24] proposed a method for automatically generating CSC corpus based on optical character recognition (COR) and automatic speech recognition (ASR). They implemented a supervised bidirectional long short-term memory (LSTM) model that turned Chinese spelling questions into a sequence tag problem. Li et al. [25] proposed a nested recurrent neural network (RNN) model for misspelling correction and trained the model using pseudo data generated from speech similarity. Liao et al. [26] proposed a Bi-LSTM-CRF generation sequence model in Chinese grammatical error diagnosis (CGED) to CGED into a sequence tag problem. In the CGED task, Zhou et al. [27] proposed a traditional linear CRF model and an LSTM-CRF model that add combination features such as positional features and syntactic features. Li et al. [28] proposed a sequence labeling method based on the strategy gradient LSTM model for the traditional model facing the imbalance of positive and negative samples and the disappearance of gradient. They used a strategy-based deep reinforcement learning approach in model training. Ren et al. [29] proposed a convolution-based sequence-to-sequence model in the Chinese grammar correction task of The Seventh CCF International conference on Natural Language Processing and Chinese Computing. They combine the advantages of convolutional neural networks and sequence-to-sequence models to treat GEC tasks as translating erroneous Chinese into correct Chinese and using CNN to capture local word sequences.

In the actual application of speech to text, we find that there are many errors in the text after speech recognition, which brings great difficulties to the text processing. This paper combines the advantages of traditional machine learning and deep learning method to correct the errors in the text converted from speech, so as to facilitate further text processing. The common text processing technology after speech recognition usually puts the error detection and correction of the speech text into the same model. In this paper, we divide the post-speech text processing task into two sub-tasks: speech text error detection and speech text error correction. In addition, we train different models separately. In the text error detection subtask, we divide the text characters to be detected into 0 and 1 categories, 0 means that the character is correct, 1 means the character is wrong, and then combined with LSTM and CRF [30] technology to train the Bi-LSTM model to handle the text error detection task. In the text error correction subtask, we first construct the replacement character set of the wrong character, then construct the Bi-LSTM model, input the replacement character of the wrong character detected in the text detection subtask into the Bi-LSTM model, and output the optimal corrected character.

The rest of the paper is organized as follows: Section 2 will elaborate on the text processing technology after speech recognition we use from two aspects: text error detection and text error correction. Section 3 will present experiments on text error detection and text error correction sub-tasks and describe the experimental results in detail. We will discuss and summarize the experimental conclusions in Sections 4 and 5.

2. Materials and Methods

In our two sub-tasks, the text error correction model relies on the detection result of the text error detection model, and the text error correction model corrects the error position word output by the text error detection model. The specific flowchart of the process is shown in Figure 1.

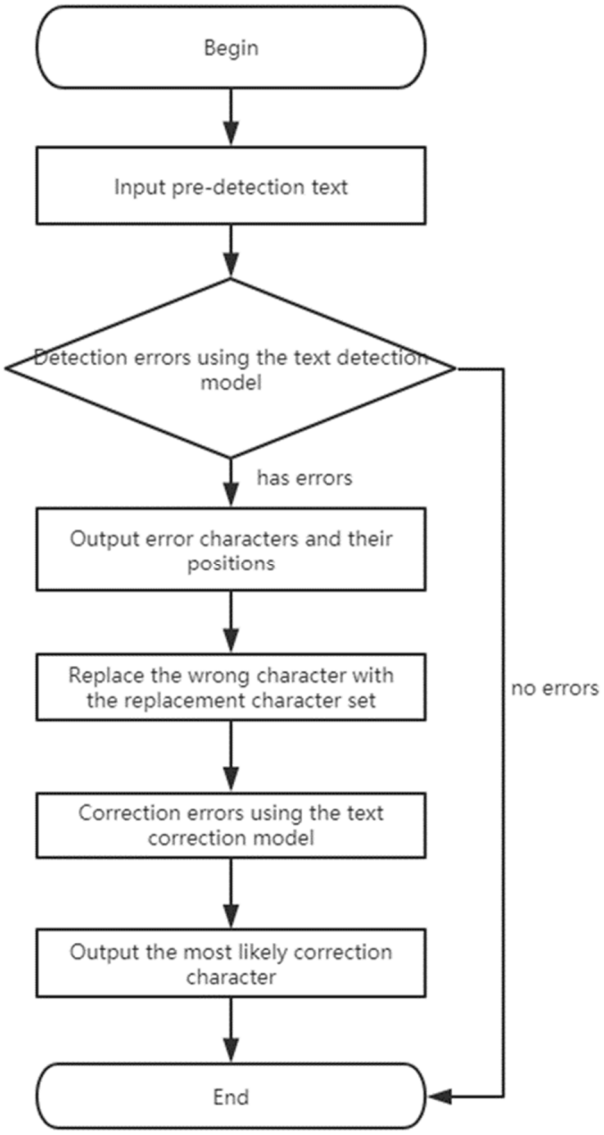

Figure 1. Speech text error detection and error correction flowchart.

In this section, we will elaborate on the model we use in the two subtasks of text error detection and text correction.

2.1. Text Error Detection

Text error detection is to mark out the errors of homophones and similar characters in the text.

In order to improve the accuracy of Chinese text error detection, we combined the advantages of the LSTM neural network model [31] and CRF technology [32], and proposed our Chinese error detection model BLSTM-CRF. The model consists of three layers: an embedded layer, a bidirectional LSTM layer, and a CRF layer. The model structure is shown in Figure 2. As can be seen from Figure 2, the text sentence is converted into a word vector matrix by embedding the layer, and then the feature matrix is extracts by the Bi-LSTM layer, and finally outputs the category (0 or 1) to which each character in the input statement belongs through the CRF layer. For example, the input sentence as shown in Figure 2 is first converted into a word vector matrix [0.1 0.2 0.3] ... [0.4 0.5 0.6], then the feature matrix is extracted through the Bi-LSTM layer, and finally the category list is output through the CRF layer [0 0 0 0 1 0 0], where 1 in the 5th position represents the 5th word in the input statement is an incorrect character and needs to be corrected. Next we will elaborate on each layer of the model.

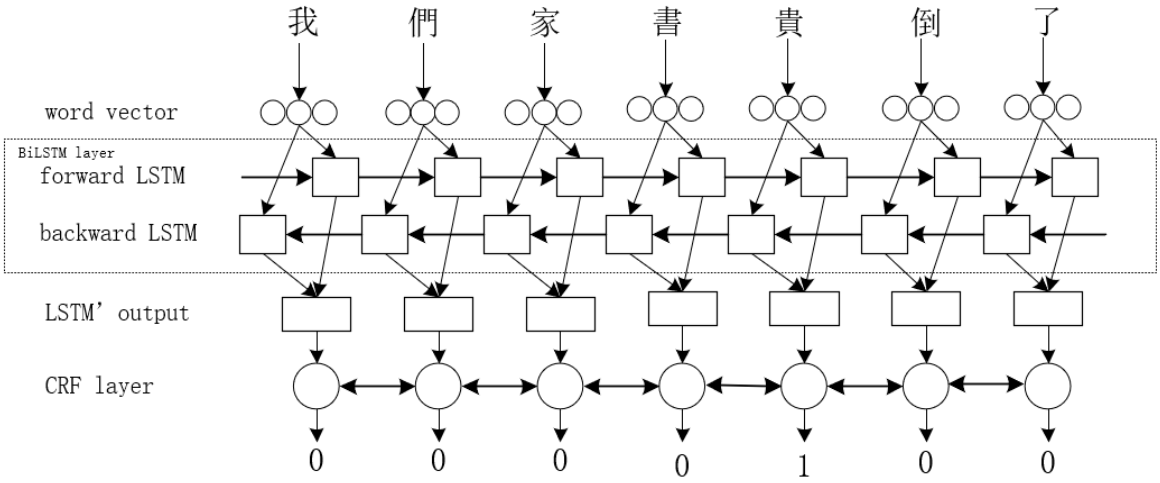

Figure 2. The structure of the text error detection model.

2.1.1. Embedded Layer

The main function of this layer is to convert characters in Chinese text into vectors.

The traditional character encoding uses the word bag model. Common methods include one-hot encoding, tf-idf encoding, and n-gram encoding. One-hot encoding [33] is to represent each character in the text in one dimension of the vector, and the text is represented by a combined long vector. If a character appears in the text, the value in the corresponding dimension is set to 1, otherwise it is 0. The advantage of this method is that the representation is simple. However, without considering the word order and semantics of the text, it is easy to suffer from data sparsity and dimension disaster. TF-IDF (word frequency-inverse document frequency) [34] considers the importance of a word in a document to the document on a one-hot basis. The importance of a word is proportional to its number of occurrences in a document, and inversely proportional to the number of occurrences in all documents. This coding method also does not consider the word order and semantics of the text. The n-gram coding method [35] considers the influence of word order in the text. The core idea is that a word in the text has a relationship with its previous n-1 words, but has no connection with other words. The advantage of this coding method is that it contains relevant information of the first n-1 words, but does not take into account the semantic relationship between different words.

In order to overcome the limitations of traditional character encoding methods, researchers have proposed the word vector coding [36]. The basic idea of word vectors is to map different words into a low-dimensional, dense vector space, and words with similar semantics map to similar positions in the word vector space. This coding method can not only reduce the representation dimension of words, but also reflect the semantic connection between different words. With the development of word vector technology, the word vector models proposed by researchers include word2vec [37], Glove [38], ELMo [39], and BERT [40].

In this paper, we use Google's BERT model to represent our word vector.

For the input text S $= [w_1, w_2, \cdots, w_i, \cdots, w_n]$, we used the BERT model to convert the input text s into a word vector matrix V $= [v_1, v_2, \cdots, v_i, \cdots, v_n]$, where w_i represents the i-th character in S, v_i represents the word vector corresponding to w_i, $V \in R^{n*d}$, d represents the dimension of the word vector.

We used the word vector matrix V as the output of the embedded layer.

2.1.2. Bi-LSTM Layer

The LSTM neural network model is a variant of the RNN [41] neural network that is suitable for processing sequence data. The neurons of the LSTM network are mainly composed of input gates, output gates, and forgetting gates. Through these three gates, the output and output information

and historical information are effectively controlled to improve the model's ability to process the sequence information.

The unit structure of the LSTM model is shown in Figure 3 [42].

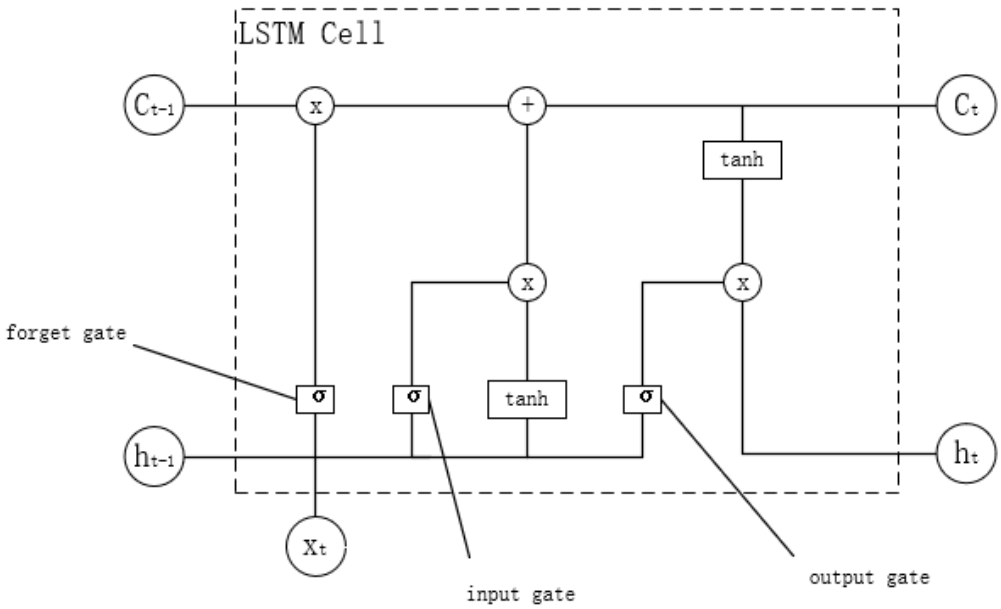

Figure 3. The unit structure of the long short-term memory (LSTM) model.

The unit is updated as follows:

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

$$i_t = \sigma(w_i \cdot [h_{t-1}, x_t] + b_i), \tag{2}$$

$$\overline{c_t} = \tanh(w_c \cdot [h_{t-1}, x_t] + b_c), \tag{3}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \overline{c_t}, \tag{4}$$

$$o_t = \sigma(w_o \cdot [h_{t-1}, x_t] + b_o), \tag{5}$$

$$h_t = o_t \cdot \tan \mathrm{h}(c_t), \tag{6}$$

where w_f, w_i, w_c, w_o represent the weight matrix, b_f, b_i, b_c, b_o represent the offset, h_{t-1} represents the hidden layer state when the time step is t, h_t represents the hidden layer state of the previous time of the time step t, x_t represents the input when the time step is t.

In the field of text information processing, the LSTM neural network model can effectively extract the context features in the input text. Usually in text data, the character before and after a word will have a certain impact on the word, so we use the bidirectional LSTM (Bi-LSTM) model to extract the contextual features of the input text.

The Bi-LSTM [43] model consists of forward LSTM and reverse LSTM, which are used to extract forward and reverse context features respectively. For the input x_t at time t, the hidden states obtained by the forward LSTM and the reverse LSTM are h'_t and h''_t, respectively.

$$h'_t = \overrightarrow{LSTM}\left(x_t, h'_{t-1}\right), \tag{7}$$

$$h''_t = \overleftarrow{LSTM}\left(x_t, h''_{t-1}\right). \tag{8}$$

The combination h_t' and h_t'' is $h_t = \left[h_t'; h_t''\right]$ as the output of the LSTM model neuron hidden layer at time t, where h_t', $h_t'' \in R^k$, $h_t \in R^{2k}$, k represents the output dimension of LSTM.

In our model, the time step is equal to the length of the input statement (t = n).

2.1.3. CRF Layer

Since the output of the Bi-LSTM layer is only one of the maximum probabilistic tag outputs in each time step, the current tag selection is still unaffected by other tags, and the outputs are independent of each other, which reduces the accuracy of the output tags. In fact, the choice of current tags is still closely related to the surrounding tags. Therefore, a CRF layer is connected behind the Bi-LSTM layer. The CRF uses the global normalization method to avoid the local optimum phenomenon in the detection result by using the feature weights and the global optimal eigenvalues with the state change, so as to improve the accuracy of the model.

CRF [44] is a model that outputs a conditional probability distribution of a random variable with a given random variable as input. For the input sequence $\mathrm{X} = (x_1, x_2, \cdots, x_n)$, labeled $\mathrm{Y} = (y_1, y_2, \cdots, y_n)$, where $y_i \in \{0,\ 1\}$, construct a linear chain conditional random field model:

$$\mathrm{G(X,\ Y)} = \sum_{i=0}^{n} T_{y_i,\ y_{i+1}} + \sum_{i=1}^{n} P_{i,\ y_i}, \tag{9}$$

$$\mathrm{P(Y|X)} = \frac{e^{G(X,\ Y)}}{\sum_{y \in Y} e^{G(X,y)}}, \tag{10}$$

where $\mathrm{T} \in R^{2*2}$ p is the probability that the BiLSTM output at the i-th position is y, and T is the transition probability from y_i to y_{i+1}, n is the length of the input sequence X.

The output of CRF [45] layer is a sequence containing 0 and 1, where the position of output 0 represents the position of the word is correct, and the position of output 1 represents the position of the corresponding character may have errors, which need to be corrected.

2.1.4. Model Training

In this model, we need to maximize the probability $\mathrm{P(Y|X)}$, so we use a log-likelihood function. The loss function we defined is as follows:

$$\mathrm{L} = \log \sum_{y \in Y} e^{G(X,y)} - G(X, Y). \tag{11}$$

In the parameter training process, our optimization algorithm uses stochastic gradient descent (SGD).

2.2. Text Error Correction

In the position where the text is detected in Section 2.1, the homonym and the near-word of the wrong character are replaced, and the replaced text is input into the Bi-LSTM evaluation model, and the replacement character with the largest output probability of the model is taken as the result of our text correction.

The text error correction model consists of two layers: the embedded layer and the Bi-LSTM layer. The model structure is shown in Figure 4. As can be seen from Figure 4, the text sentence after the replacement character is first input into the embedded layer and converted into a word vector matrix, and then the Bi-LSTM layer outputs the probability that the replacement character is the correct character. For example, replacing the 5th word in Figure 2 with the 5th word as shown in Figure 4 in the text statement, the probability of outputting to the correct character after inputting the Bi-LSTM layer is 0.328.

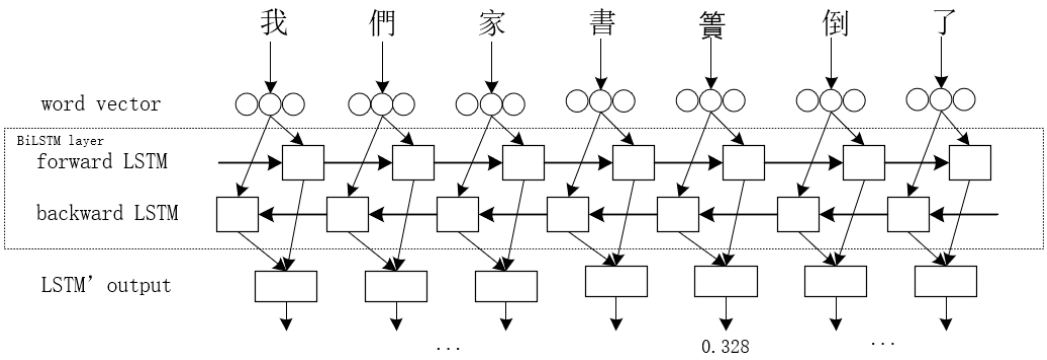

Figure 4. The structure of the text error correction model.

2.2.1. Constructing a Replacement Character Set

The replacement character set consists of homophonic characters and near-characters provided by the SIGHAN7 dataset [46]. We combine the homophone and the near-characters of each character as its replacement character list. Some examples of the replacement character list are shown in Figure 5.

The character	Replacement characters
七	悽淒柒妻欺戚棲溪萋谿漆慼
乙	尾椅以蟻已矣旖偯迤迆倚
八	把跋靶伯吧杷霸耙壩鈸爸罷拔琶瀟

Figure 5. The example of the replacement character set.

2.2.2. Embedded Layer

The structure and processing of this layer are similar to those in Section 2.1.1. We use the replacement character table of the error character to construct their word vector separately by referring to Section 2.1.1.

2.2.3. Bi-LSTM Layer

This layer structure is similar to the Bi-LSTM structure described in Section 2.1.2. For different replacement characters, the word vector input in Section 2.2.2 will have different probabilities after passing through the Bi-LSTM layer of this layer. We select the replacement character with the largest output probability of Bi-LSTM layer as the result of the text error correction model from the candidate replacement characters.

If there is a candidate set $C = (c_1, c_2, \cdots, c_n)$ of erroneous words, after the BiLSTM layer is followed by the probability set $P = (p_1, p_2, \cdots, p_n)$, we select the character corresponding to max(P) as the result of the text error correction model.

2.2.4. Model Training

To get the probability that the output is of a certain class, we use the cross entropy loss function.

$$\mathrm{L} = -\sum_{i=1}^{n} y_i \log p_i, \tag{12}$$

where n represents the number of types ($n = 2$), y_i represents the indicator variable (0 or 1) of the category, and p_i represents the predicted probability that the sample belongs to category i.

In the parameter training process, our optimization algorithm uses Adam. The Adam algorithm is a first order optimization algorithm with adaptive learning rate that allows the model to converge quickly.

3. Results

In this section, we evaluate the performance of our proposed model on text error detection and correction. Section 3.1 introduces the datasets we used in our experiments, Section 3.2 describes the performance metrics we used, Section 3.3 describes the data preprocessing process we performed on the dataset, and Section 3.4 describes the results of our model and compares it with other models.

3.1. Datasets

Our experiment uses SIGHAN 2013 CSC Datasets. The test set for this dataset provides a sample set and a similar character set. The sample set consists of 700 samples from student articles, half of which contain at least one error and the remaining samples do not contain any errors. Similar character sets contain characters that are similar in shape to Chinese characters and sound similar in pronunciation. The test set is composed of students from 13 to 14 years old. In the text error detection task, the test set contains 1000 test sentences, of which 300 have errors. In a text correction task, the test set contains 1000 test sentences, each of which contain one or more errors.

3.2. Performance Metrics

3.2.1. Metrics of Error Detection

We use the following indicators to evaluate the performance of the text detection model:

(1) Detection Accuracy (DA): The number of sentences with the correct detection result / the total number of sentences.
(2) Detection Precision (DP): The actual number of errors detected / the number of error sentences detected.
(3) Detection Recall (DR): The actual number of errors detected / the number of actual error sentences.
(4) Detection F1 (DF1): 2 * DP * DR / (DP + DR).

3.2.2. Metrics of Error Correction

We use the following indicators to evaluate the performance of the text correction model:

(1) Location Accuracy (LA): The number of sentences correctly detected the error location / The total number of sentences.
(2) Correction Accuracy (CA): The number of sentences correctly corrected the error / The total number of sentences.
(3) Correction Precision (CP): The number of sentences correctly corrected the error / The number of error sentences detected by the system.

3.3. Data Preprocessing

(1) Clear all punctuation in the sentence, leaving only plain text data.
(2) The original data text is marked according to the character level model, the correct word is marked as 0, the wrong word is marked as 1, and the obtained 0, 1 sequence is used as the label of the original data text.
(3) Calculate the number of different characters in the dataset after preprocessing, the number of occurrences of each character, the length of the largest character contained in each sentence, and the average character length of each sentence. The maximum sentence length is used as a fixed length for characters in each sentence. If the length of the sentence is greater than the fixed length, it is truncated. If the length of the sentence is less than the fixed length, 0 is added.

3.4. Experimental Results

Tensorflow is Google's open source data flow diagram-based machine learning framework, keras is for tensorflow, and Theano and high-level encapsulation. People can quickly build a complex neural network model by using them. Therefore, we used tensorflow and keras to implement the model we described in Section 2.

To more accurately evaluate the performance of our proposed model, we used a 10-fold cross-validation [47] and 5*2 cross-validation [48] approach to partition the dataset we used. In the 10-fold cross-validation method, we randomly divided the dataset into 10 parts, using nine of them as the training set in turn, and the remaining part as the verification set, taking the mean of these 10 results as the evaluation result of our model. In the 5*2 cross-validation method, we randomly divided the dataset into five parts, used four of them as the training set in turn, and the remaining part as the test set, and in the data used as the training set, we used the 2-fold cross-validation for training, and the average of the five results was taken as the evaluation result of our model. Tables 1 and 2 show the classification results in the text error detection model, respectively. Tables 3 and 4 show the classification results in the text error correction model, respectively.

Table 1. The 10-fold cross-validation results of text error detection model.

Dataset	DA	DP	DR	DF1
Dataset1	0.901	0.866	0.893	0.879
Dataset2	0.886	0.85	0.896	0.872
Dataset3	0.893	0.851	0.903	0.876
Dataset4	0.908	0.871	0.886	0.878
Dataset5	0.91	0.875	0.912	0.893
Dataset6	0.889	0.862	0.908	0.884
Dataset7	0.90	0.863	0.892	0.877
Dataset8	0.882	0.848	0.873	0.860
Dataset9	0.889	0.852	0.912	0.880
Dataset10	0.902	0.859	0.874	0.866
Average	0.896	0.859	0.894	0.876

Table 2. The 5*2 cross-validation results of text error detection model.

Dataset	DA	DP	DR	DF1
Dataset1	0.936	0.893	0.915	0.903
Dataset2	0.934	0.886	0.898	0.891
Dataset3	0.946	0.892	0.917	0.904
Dataset4	0.931	0.904	0.901	0.902
Dataset5	0.942	0.898	0.899	0.898
Average	0.937	0.894	0.906	0.9

Table 3. The 10-fold cross-validation results of text error correction model.

Dataset	LA	CA	CR
Dataset1	0.742	0.706	0.782
Dataset2	0.726	0.712	0.773
Dataset3	0.728	0.698	0.769
Dataset4	0.736	0.692	0.789
Dataset5	0.729	0.702	0.766
Dataset6	0.735	0.699	0.791
Dataset7	0.736	0.705	0.788
Dataset8	0.733	0.704	0.783
Dataset9	0.74	0.713	0.776
Dataset10	0.739	0.704	0.781
Average	0.734	0.703	0.779

Table 4. The 5*2 cross-validation results of text error correction model.

Dataset	LA	CA	CR
Dataset1	0.784	0.733	0.817
Dataset2	0.791	0.734	0.82
Dataset3	0.785	0.74	0.813
Dataset4	0.779	0.733	0.81
Dataset5	0.784	0.731	0.816
Average	0.784	0.734	0.815

It is well known that the number of iterations of the model affects the final performance of the model. We experimented with our models using different iterations. As the number of iterations increases, the model will gradually over-fit, resulting in a decrease in generalization performance as shown in Tables 5 and 6. When the number of iterations is less than 10 times, the performance of the model gradually increases. When the number of iterations of the model is more than 10 times, the performance of the model gradually decreases with the increase of the number of iterations.

Table 5. The effect of the number of iterations on the text error detection model.

Epochs	DA	DP	DR	DF1
5	0.886	0.861	0.871	0.865
8	0.892	0.869	0.876	0.872
10	0.901	0.866	0.893	0.879
12	0.896	0.861	0.881	0.870
15	0.889	0.857	0.876	0.866

Table 6. The effect of the number of iterations on the text error correction model.

Epochs	LA	CA	CR
5	0.733	0.689	0.768
8	0.740	0.710	0.776
10	0.742	0.706	0.782
12	0.736	0.698	0.781
15	0.731	0.695	0.773

In the field of natural language processing, the input text statements are usually of different lengths. Unifying the length of the input statements to different lengths also affects the performance of the network model. We unified the length of the input statement to the length of the largest sentence in the dataset and the average sentence length, and experimented on these two methods. Tables 7 and 8 show the experimental results using the length of these two input statements. Using the average sentence length will intercept the statement in the dataset whose statement length is longer than the average sentence length, resulting in the missing part of the context information of these sentences, affecting the final experimental results of the model. As can be seen from Tables 7 and 8, using the maximum sentence length in the dataset as the fixed length of the input sentence can result in better experimental results than using the average sentence length.

Table 7. The effect of the fixed length of the input statement on the text error detection model.

SENTENCE_LENGTH	DA	DP	DR	DF1
37	0.886	0.853	0.871	0.861
104	0.901	0.866	0.893	0.879

Table 8. The effect of the fixed length of the input statement on the text error correction model.

SENTENCE_LENGTH	LA	CA	CR
37	0.726	0.698	0.768
104	0.742	0.706	0.782

We compared our model with the experimental results of other models. The results of the error detection experiments of our model on the SIGHAN 2013 CSC dataset are shown in Table 9 and Figure 6.

Table 9. Text error detection model experimental results.

Model	DA	DP	DR	DF1
CRF	0.722	0.6964	0.13	0.2191
CKIP-WS	0.84	0.7174	0.77	0.7428
CKIP-WS + G1-WS	0.842	0.6919	0.8533	0.7642
BiLSTM-CRF	0.89	0.87	0.93	0.91

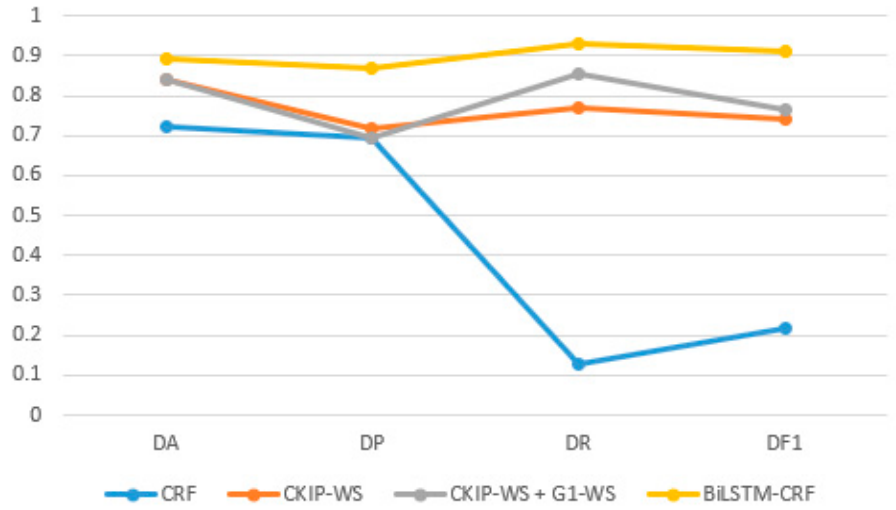

Figure 6. Text error detection model experimental results comparison chart.

In the method proposed by Hsieh et al. [49], in the CKIP-WS system, they first segment the Chinese sentences and then search for possible replacement words through the confusion set and the CKIP dictionary. In the G1-WS system, after the Chinese sentence segmentation, the similar words are searched by the dictionary constructed by Google's unigram data, and then sorted according to their frequencies, and the low frequency words are used as the wrong candidates. The error detection performance of the two method models is overly dependent on the richness of the dictionary and confusion set they construct, and has greater limitations in different fields. In the method proposed by Wang et al. [50] they first use a rule-based way to check for high-frequency erroneous words in the sentence, and then use a CRF-based parser to divide the sentence into a sequence of words. The characters in each short word (less than three characters) are considered to be potential erroneous characters. The performance of their error detection depends on the detection rules they construct. Compared to their models, we can use the LSTM neural network to automatically extract character features that are more complex than them, without the need to build a dictionary or rule, and use the transfer feature of the CRF model to correlate the outputs of the LSTM network to obtain more better error detection performance.

The results of the error correction experiments of our model on the SIGHAN 2013 CSC dataset are compared with other models as shown in Table 10 and Figure 7.

Table 10. Text error correction model experimental results.

Model	LA	CA	CR
CRF	0.485	0.404	0.404
CKIP-WS	0.482	0.442	0.5854
CKIP-WS + G1-WS	0.559	0.516	0.6158
BiLSTM	0.75	0.70	0.72

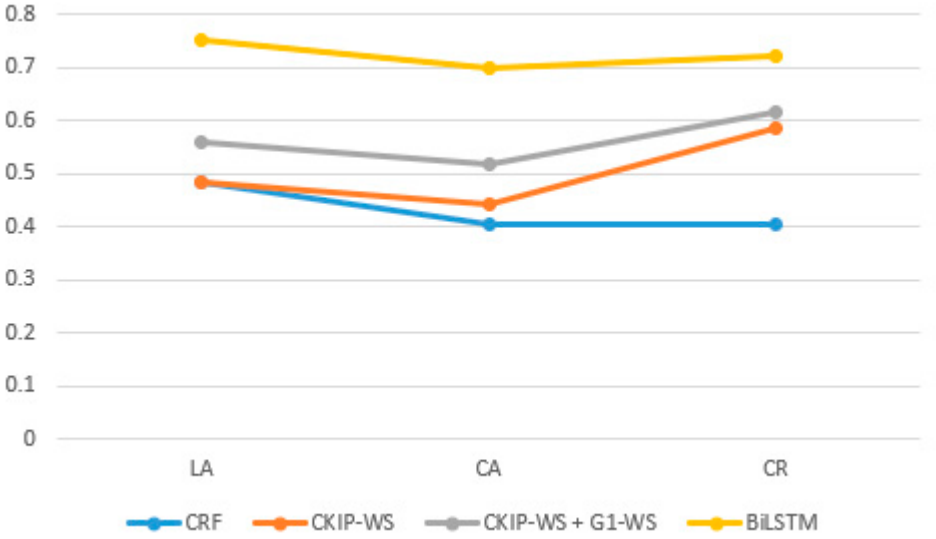

Figure 7. Text error correction model experimental results comparison chart.

In the method of Hsieh et al., they use the similar characters of the wrong character in the confusion set or dictionary as candidate characters, and then construct a 3-gram model to determine the best character sequence as the correction result. In the method of Wang et al., they replace the detected wrong characters with characters of the same shape or similar pronunciation, and then use the CRF parser to re-segment the modified sentences and use the LM model to score, and take the sentence with the highest LM score as the correction result. Compared with their models, we use the Bi-LSTM model to get more context features than them, so as to get better error correction results.

4. Discussion

In this paper, we proposed a new text processing technology for speech recognition. We divided the text processing tasks for speech recognition into two sub-tasks: speech text error detection and speech text error correction. In the speech text error detection subtask, we developed a Bi-LSTM-CRF model based on the advantages of the LSTM neural network and the CRF model. The LSTM model is a variant of the RNN model. Its neurons can effectively retain the historical information in the input sequence through the input gate, output gate, and forget gate, which is very suitable for processing sequence data. For text data, each word has a certain relationship with its before and after words, so the LSTM neural network model can effectively extract the context features in the text data and better process the text data. The bidirectional LSTM neural network model we used can simultaneously obtain the forward and backward dependence features of a word in a sentence, which can better reflect the characteristics of the input text. Compared with the n-gram method proposed by Hsieh et al., the n-gram model is a statistical language model whose core assumption is that the probability of occurrence of the current word in the text sentence is only related to the previous n-1 words. As n increases, the more dependent features of the obtained words will be, the prediction of future words will be more accurate, but when n increases, sparse problems will occur, which will affect the accuracy of future word prediction. The Bi-LSTM network we used not only can obtain long-distance forward text features, but also obtain backward text features. The output of the separate Bi-LSTM model does

not take into account the contextual relationship between the outputs and may result in impossible results in the locale, so we introduced the CRF model. The CRF model can relate the output of the Bi-LSTM model to each other, which is equivalent to treating the output as a 2^n classification problem, which can effectively eliminate some impossible results and output more accurate results. The experimental results in Section 3 show that our model has better text error detection performance than the n-gram model used by Hsieh et al. and the CRF model used by Wang et al.

In the speech text error correction subtask, we first constructed the candidate character set of the wrong character, and then input the candidate characters of the wrong character detected in the speech text error detection subtask into the Bi-LSTM neural network, and selected the candidate character with the largest output probability. As our correct character. Compared with other statistical probability models, the Bi-LSTM model we used can better extract the context features between input texts, and then selected the optimal correction characters by comparing different error candidate characters. The experimental results in Section 3 show that our proposed text error correction model has better text error correction performance with other text error correction models.

In addition, we also explored how the performance of the model changes with the number of iterations as the number of iterations of the model changes. By analyzing the experimental results, we find that when the number of iterations of the model increases slowly, the performance of the model will gradually become better, but when the number of iterations reaches a certain value, the performance of the model will gradually decrease. As the number of model iterations is very large, although the performance on the training set is very effective, the model parameters over-fit the training set data, resulting in poor generalization performance of the model, affecting the effect of the model on the test set. In addition, the text length of the input model will also have an impact on the model's effects. We chose the maximum sentence length in the dataset and the length of the average sentence as the fixed input length of the model. The experimental results show that the fixed input length of the model has a better effect when the maximum sentence length is taken. Since the input length can take more text features when the maximum sentence length is obtained, a better effect model can be obtained.

5. Conclusions and Future Work

The development of speech recognition technology is closely related to the progress of wireless networks. With the rapid development of wireless network technology [51–56], speech recognition technology will be more and more widely used in the field of Internet of Things. Due to the problems of dialect and accent, it is necessary to correct the text after speech recognition before displaying. However, text processing technology for speech recognition has been attracted more and more attention of researchers. This paper proposes a text processing model after Chinese speech recognition based on a LSTM neural network model and CRF technology. The model can fully extract the context information of the input text. Through the analysis of the experimental results, we found that compared with other models, our proposed neural network model has better text error detection and correction performance.

In addition, by increasing the amount of data in the replacement set, the error detection and correction performance of the model can be further improved.

In this article, raw data was processed by a character-level model. Given the advantages of the word-level model, we can try to process the raw data using a word-level model. In addition, in the phonetic text, in addition to homophones and approximations, there may be grammatical errors, which can be used as our next research direction.

Author Contributions: Conceptualization, L.Y.; Data curation, Y.L.; Formal analysis, Z.T.; Methodology, J.W.; Project administration, Z.T.; Supervision, J.W.; Writing—original draft, Y.L.; Writing—review and editing, L.Y.

Funding: This research was funded by the National Natural Science Foundation of China [Nos. 61602060, 61772454, 61811530332, 61811540410], the Open Research Fund of Hunan Provincial Key Laboratory of Intelligent Processing of Big Data on Transportation [2015TP1005].

Acknowledgments: We are grateful to our anonymous referees for their useful comments and suggestions. The authors also thank Keqin Li, Ying Zheng for their useful advice during this work.

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

References

1. Wang, J.; Gao, Y.; Liu, W.; Wu, W.; Lim, S.J. An asynchronous clustering and mobile data gathering schema based on timer mechanism in wireless sensor networks. *Comput. Mater. Contin.* **2019**, *58*, 711–725. [CrossRef]
2. Wang, J.; Ju, C.W.; Gao, Y.; Sangaiah, A.K.; Kim, G. A PSO based energy efficient coverage control algorithm for wireless sensor networks. *Comput. Mater. Contin.* **2018**, *56*, 433–446.
3. Wang, J.; Gao, Y.; Yin, X.; Li, F.; Kim, H.J. An enhanced PEGASIS algorithm with mobile sink support for wireless sensor networks. *Wirel. Commun. Mob. Comput.* **2018**, *2018*. [CrossRef]
4. Pan, J.S.; Kong, L.P.; Sung, T.W.; Tsai, P.W.; Snasel, V. Alpha-Fraction First Strategy for Hierarchical Wireless Sensor Networks. *J. Internet Technol.* **2018**, *19*, 1717–1726.
5. Pan J, S.; Lee, C.Y.; Sghaier, A.; Zeghid, M.; Xie, J.F. Novel Systolization of Subquadratic Space Complexity Multipliers Based on Toeplitz Matrix-Vector Product Approach. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2019**, *27*, 1614–1622. [CrossRef]
6. Nguyen, T.T.; Pan, J.S.; Dao, T.K. An Improved Flower Pollination Algorithm for Optimizing Layouts of Nodes in Wireless Sensor Network. *IEEE Access* **2019**, *7*, 75985–75998. [CrossRef]
7. Neumann, L.; Matas, J. Real-time lexicon-free scene text localization and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 1872–1885. [CrossRef]
8. Tang, Z.; Ding, X.F.; Zhong, Y.; Yang, L.; Li, K.Q. A Self-Adaptive Bell–LaPadula Model Based on Model Training with Historical Access Logs. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2047–2061. [CrossRef]
9. Chiu, H.; Wu, J.; Chang, J.S. Chinese spelling checker based on statistical machine translation. In Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing, Nagoya, Japan, 14–18 October 2013; pp. 49–53.
10. Yeh, J.F.; Chang, L.T.; Liu, C.Y.; Hsu, T.W. Chinese spelling check based on N-gram and string matching algorithm. In Proceedings of the 4th Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2017), Taipei, Taiwan, 1 December 2017; pp. 35–38.
11. Hasan, S.; Heger, C.; Mansour, S. Spelling correction of user search queries through statistical machine translation. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 451–460.
12. Hsieh, Y.M.; Bai, M.H.; Huang, S.L.; Chen, K.J. Correcting Chinese spelling errors with word lattice decoding. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2015**, *14*, 18. [CrossRef]
13. Siklósi, B.; Novák, A.; Prószéky, G. Context-aware correction of spelling errors in Hungarian medical documents. *Comput. Speech Lang.* **2016**, *35*, 219–233. [CrossRef]
14. Huang, Q.; Huang, P.J.; Zhang, X.R.; Xie, W.J.; Hong, K.D.; Chen, B.Z.; Huang, L. Chinese spelling check system based on tri-gram model. In Proceedings of the Third CIPS-SIGHAN Joint Conference on Chinese Language Processing, Wuhan, China, 20–21 October 2014; pp. 173–178.
15. Han, D.; Chang, B. A maximum entropy approach to Chinese spelling check. In Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing, Nagoya, Japan, 14–18 October 2013; pp. 74–78.
16. Zhao, H.; Cai, D.; Xin, Y.; Wang, Y.Z.; Jia, Z.Y. A hybrid model for Chinese spelling check. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2017**, *16*, 21. [CrossRef]
17. Chen, K.Y.; Wang, H.M.; Chen, H.H. A probabilistic framework for Chinese spelling check. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2015**, *14*, 15. [CrossRef]
18. Liu, X.; Cheng, F.; Duh, K.; Matsumoto, Y.J. A hybrid ranking approach to Chinese spelling check. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2015**, *14*, 16. [CrossRef]
19. Yeh, J.F.; Chen, W.Y.; Su, M.C. Chinese spelling checker based on an inverted index list with a rescoring mechanism. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2015**, *14*, 17. [CrossRef]
20. He, S.M.; Li, Z.Z.; Tang, Y.N.; Liao, Z.F.; Wang, J.; Kim, H.J. Parameters Compressing in Deep Learning. *Comput. Mater. Contin.* **2019**, accepted.
21. Zeng, D.J.; Dai, Y.; Li, F.; Wang, J.; Sangaiah, A.K. Aspect based sentiment analysis by a linguistically regularized CNN with gated mechanism. *J. Intell. Fuzzy Syst.* **2019**, *36*, 3971–3980. [CrossRef]
22. Tu, Y.; Lin, Y.; Wang, J.; Kim, J.U. Semi-supervised learning with generative adversarial networks on digital signal modulation classification. *Comput. Mater. Contin.* **2018**, *55*, 243–254.

23. Zeng, D.J.; Dai, Y.; Li, F.; Sherratt, R.S.; Wang, J. Adversarial learning for distant supervised relation extraction. *Comput. Mater. Contin.* **2018**, *55*, 121–136.
24. Wang, D.M.; Song, Y.; Li, J.; Han, J.L.; Zhang, H.S. A Hybrid Approach to Automatic Corpus Generation for Chinese Spelling Check. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2517–2527.
25. Li, H.; Wang, Y.; Liu, X.Y.; Sheng, Z.C.; Wei, S. Spelling Error Correction Using a Nested RNN Model and Pseudo Training Data. *arXiv* **2018**, arXiv:1811.00238.
26. Liao, Q.L.; Wang, J.; Yang, J.N.; Zhang, X.J. Ynu-hpcc at ijcnlp-2017 task 1: Chinese grammatical error diagnosis using a bi-directional lstm-crf model. In Proceedings of the IJCNLP 2017, Shared Tasks, Taipei, Taiwan, 27 November–1 December 2017; pp. 73–77.
27. Zhou, Y.; Shao, Y.; Zhou, Y. Chinese Grammatical Error Diagnosis Based on CRF and LSTM-CRF model. In Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications, Melbourne, Australia, 19 July 2018; pp. 165–171.
28. Li, C.; Qi, J. Chinese Grammatical Error Diagnosis Based on Policy Gradient LSTM Model. In Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications, Melbourne, Australia, 19 July 2018; pp. 77–82.
29. Ren, H.; Yang, L.; Xun, E. A Sequence to Sequence Learning for Chinese Grammatical Error Correction. In Proceedings of the CCF International Conference on Natural Language Processing and Chinese Computing, Hohhot, China, 26–30 August 2018; pp. 401–410.
30. Yang, L.; Zhou, Y.; Zheng, Y. Annotating the literature with Disease Ontology. *Chin. J. Electron.* **2017**, *26*, 1261–1268. [CrossRef]
31. Makarenkov, V.; Guy, I.; Hazon, N.; Meisels, T.; Shapira, B.; Rokach, L. Implicit dimension identification in user-generated text with LSTM networks. *Inf. Process. Manag.* **2019**, *56*, 1880–1893. [CrossRef]
32. Yang, L.; Wang, J.; Tang, Z.; Xiong, N. Using Conditional Random Fields to Optimize a Self-Adaptive Bell-LaPadula Model in Control Systems. *IEEE Trans. Syst. Manand Cybern. Syst.* **2019**, 1–15. [CrossRef]
33. Wang, P.; Xu, B.; Xu, J.; Tian, G.; Liu, C.L. Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification. *Neurocomputing* **2016**, *174*, 806–814. [CrossRef]
34. Chen, K.; Zhang, Z.; Long, J.; Zhang, H. Turning from TF-IDF to TF-IGM for term weighting in text classification. *Expert Syst. Appl.* **2016**, *66*, 245–260. [CrossRef]
35. Tripathy, A.; Agrawal, A.; Rath, S.K. Classification of sentiment reviews using n-gram machine learning approach. *Expert Syst. Appl.* **2016**, *57*, 117–126. [CrossRef]
36. Ma, Z.; Li, M. Chinese Text Similarity Algorithm Based on Part-of-Speech Tagging and Word Vector Model. *JCP* **2019**, *14*, 311–317.
37. Li, W.; Zhu, L.; Guo, K.; Shi, Y.; Zheng, Y. Build a tourism-specific sentiment Lexicon via word2vec. *Ann. Data Sci.* **2018**, *5*, 1–7. [CrossRef]
38. Li, Y.; Pan, Q.; Yang, T.; Wang, S.; Tang, J.; Cambria, E. Learning word representations for sentiment analysis. *Cogn. Comput.* **2017**, *9*, 843–851. [CrossRef]
39. Peters, M.E.; Neumann, N.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. *arXiv* **2018**, arXiv:1802.05365.
40. Sun, C.; Huang, L.; Qiu, X. Utilizing BERT for Aspect-Based Sentiment Analysis via Constructing Auxiliary Sentence. *arXiv* **2019**, arXiv:1903.09588.
41. Wei, D.; Wang, B.; Lin, G.; Liu, D.; Dong, Z.; Liu, H.; Liu, Y. Research on unstructured text data mining and fault classification based on RNN-LSTM with malfunction inspection report. *Energies* **2017**, *10*, 406. [CrossRef]
42. Yao, Y.S.; Huang, Z. Bi-directional LSTM recurrent neural network for Chinese word segmentation. In Proceedings of the International Conference on Neural Information Processing, Kyoto, Japan, 16–21 October 2016.
43. Liang, D.; Liang, H.; Yu, Z.; Zhang, Y. Deep convolutional BiLSTM fusion network for facial expression recognition. *Vis. Comput.* **2019**. [CrossRef]
44. Li, T.T.; Ji, D.H. Sentiment analysis of micro-blog based on SVM and CRF using various combinations of features. *Appl. Res. Comput.* **2015**, *32*, 978–981.

45. Chen, T.; Xu, R.; He, Y.; Wang, X. Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN. *Expert Syst. Appl.* **2017**, *72*, 221–230. [CrossRef]
46. Wu, S.H.; Liu, C.L.; Lee, L.H. Chinese spelling check evaluation at SIGHAN Bake-off 2013. In Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing, Nagoya, Japan, 14–18 October 2013; pp. 35–42.
47. Jung, Y.; Hu, J. AK-fold averaging cross-validation procedure. *J. Nonparametric Stat.* **2015**, *27*, 167–179. [CrossRef]
48. Dora, L.; Agrawal, S.; Panda, R.; Abraham, A. Nested cross-validation based adaptive sparse representation algorithm and its application to pathological brain classification. *Expert Syst. Appl.* **2018**, *114*, 313–321. [CrossRef]
49. Hsieh, Y.M.; Bai, M.H.; Chen, K.J. Introduction to CKIP Chinese spelling check system for SIGHAN Bakeoff 2013 evaluation. In Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing, Nagoya, Japan, 14–18 October 2013; pp. 59–63.
50. Wang, Y.R.; Liao, Y.F.; Wu, Y.K.; Chang, L.C. Conditional Random Field-based Parser and Language Model for Traditional Chinese Spelling Checker. In Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing, Nagoya, Japan, 14–18 October 2013; pp. 69–73.
51. Wang, J.; Gao, Y.; Liu, W.; Sangaiah, A.K.; Kim, H.J. Energy Efficient Routing Algorithm with Mobile Sink Support for Wireless Sensor Networks. *Sensors* **2019**, *19*, 1494. [CrossRef]
52. Wang, J.; Gao, Y.; Wang, K.; Sangaiah, A.K.; Lim, S.J. An Affinity Propagation-Based Self-Adaptive Clustering Method for Wireless Sensor Networks. *Sensors* **2019**, *19*, 2579. [CrossRef]
53. Xia, Z.Q.; Fang, Z.W.; Zou, F.F.; Wang, J.; Sangaiah, A.K. Research on Defensive Strategy of Real-Time Price Attack Based on Multiperson Zero-Determinant. *Secur. Commun. Netw.* **2019**. [CrossRef]
54. Meng, Z.; Pan, J.S.; Tseng, K.K. PaDE: An enhanced Differential Evolution algorithm with novel control parameter adaptation schemes for numerical optimization. *Knowl. Based Syst.* **2019**, *168*, 80–99. [CrossRef]
55. Pan, J.S.; Kong, L.; Sung, T.W.; Tsai, P.W.; Snášel, V. A clustering scheme for wireless sensor networks based on genetic algorithm and dominating set. *J. Internet Technol.* **2018**, *19*, 1111–1118.
56. Wu, T.Y.; Chen, C.M.; Wang, K.H.; Meng, C.; Wang, K.E. A provably secure certificateless public key encryption with keyword search. *J. Chin. Inst. Eng.* **2019**, *42*, 20–28. [CrossRef]