Journal of Computers Vol. 33 No. 1, February 2022, pp. 139-150
doi: 10.53106/199115992022023301013

An Errors Correction Model for the Errors of Non-word and Real-word in English Composition

Guimin Huang, Maolin Li*

Guangxi Key Laboratory of Image and Graphic Intelligent Processing,
School of Computer Science and Information Security,
Guilin University of Electronic Technology Guilin 541004, China
sendhuang@126.com, lml14026@163.com
Received 18 May 2021; Revised 7 September 2021; Accepted 6 October 2021

Abstract. In the procedure from composing English, it is inevitable to face the phenomenon of word writing errors. In recent years, English composition automatic correcting system has attracted much attention. However, the precision of the existing word errors correcting system is vague generalization. So as to move forward the accuracy of checking and correcting word errors, this paper designs a word errors correction model based on natural language processing technology. This model designs phoneme matching method based on an improved IDM algorithm, and combined with a non-word input errors correction method based on character distance. The accuracy of correcting non-word errors in this model reached 86.5%. The study also proposes a real-word errors correction method, which is implemented basing on the real-word confusion set and combining the binary statistical model and the GloVe word vector model, improving the real-word errors correction method based on feature annotation of the real-word confusion set, with an accuracy of 77.9%.

Keywords: English composition, IDM algorithm, binary statistical model, word embedding model

1 Introduction

In the current English education system in China, the teacher-student ratio is not enough, so it is difficult for English teachers to take care of each student thoughtfully. However, it takes a lot of practice to improve the ability of English subjective questions, and students can't get timely feedback on their practice results. Therefore, it is very necessary to research and develop an errors correction model for writing words. For written English texts, the ability to detect and correct word errors has evolved significantly in recent years, partly based on the development of artificial intelligence and the continuous improvement of natural language processing techniques [1]. Words are the basic units for expressing the semantics of the original text, and subsequent processing tasks have their roots in the processing of words. Problems with words will affect the subsequent processing and analysis of the passage, and influence the performance of the whole processing task. In studies of automatic errors correction for English words, word errors are usually classified into two types: non-word errors and real-word errors [2]. Non-word errors refer to the spelling of words as words that do not actually exist in the dictionary, which is subdivided into typographic errors and cognitive errors. In the case of typographic errors, the writer is considered to know the correct spelling but press the wrong input key; The source of cognitive errors is presumed to be a misunderstanding or a lack of knowledge on the part of the writer, and occur when the word usually has the same or similar pronunciation as the correct word. A real-word error is when students misspell a word as another similar word in the dictionary, but not the true correct word. Many word correction programs are not able to detect such word errors; they can only deal with isolated non-word errors that do not exist in the dictionary, and usually the word in which real-word errors occurs does not fit into the context of the sentence in which it occurs and appears to be very abrupt in its semantic expression. Therefore, real-word errors correction is more difficult than non-word errors [3].

The application of computers for automatic correction of word errors has a history of several decades, with an earlier spell checker for English called TYPO developed by IBM's research laboratory and a spell checker program called Spell developed by R. Gorin, which plays an important role in improving the quality of documents by identifying spelling errors in documents [4]. Spell checkers are used to identify and correct errors made by users while writing text, using contextual information about confusing words to automatically correct spelling errors in text editors or text documents [5]. Currently, GNU Aspell [6] and Hunspell [7] are widely used free word correction programs and are often used as benchmarks for word correction studies, suggesting that benchmark data sets are the preferred benchmark data sets for newly developed isolated word spell checkers. However, there is still

* Corresponding Author

room for improvement in these errors correction systems, and further research into the checking and correction of word errors is worthwhile. This paper designs an errors correction model for non-word and real-word errors in English compositions to improve their accuracy rate more effectively.

The main contributions of this paper are as follows:

The non-word errors checking function by constructing a dictionary tree, using the method of editing distance to get corresponding suggestions from the dictionary to correct non-word errors caused by operations such as mistyping, and then using modified version of IDM phoneme matching to correct non-word errors caused by cognitive errors [8].

The real-word errors correction method in view of a predefined real-word confusion set, designs a matching binary statistical model to complement the algorithm of candidate words, determines the contextual semantic relevance of words by GloVe, and improves the real-word errors correction method based on feature annotation of the real-word confusion set.

Through the analysis and validation of experiments, the accuracy rate of non-word errors correction of this model is 86.5%, and the correction accuracy of real-word errors reached 77.9%. The results show that the proposed method has good results in correcting non-word errors and real-word errors in English compositions.

2 Related Work

There are two main methods for checking non-word errors, one based on dictionary pairing and another based on N-Gram statistical analysis. The dictionary pairing approach is to pair the words to be checked with a pre-given list of dictionaries, and the main methods currently used for querying dictionaries are the binary tree, the finite-state automaton and the hashing [9]. The method based on N-Gram statistical analysis is an N-Gram partitioning of the words to be checked, where each N-Gram string in a word is matched with an N-Gram frequency table that is counted and computed from the correct corpus [10]. For the correction of non-word errors are generally only concerned with the word itself, common errors correction methods have the minimum edit distance, the similar skeleton key , the misspelling statistical dictionary, and the N-Gram statistical analysis. The minimum edit distance method [11] is to analyze the difference of strings morphological by the number of operations to transform the previous string into the later one by substitution, deletion, insertion and interchange, and calculating the minimum edit distance required for the transformation, the smaller the distance, the more likely it is to be the correct word. Similar skeleton keys are used to transform a word into a skeleton key that can represent the word by set rules, and then match strings with similar string morphology or similar pronunciation by skeleton keys [9]. The misspelling statistics dictionary-based approach uses a misspelling statistics dictionary that has been developed by manually counting a large number of texts in which non-word errors occur. Leacock et al. state that most approaches to detect and correct collocation errors use an association strength measure to compare the author's word choice with a set of alternatives and select the combination with the highest score. This strategy relies on a comparison with a set of alternatives, capacity is limited, and detection cannot be performed independently of correction [12]. The comparison of existing word errors correction methods is shown in Table 1.

Currently, the main research on correct methods of real-word errors mainly includes N-Gram statistical language models, methods based on machine learning and semantic information. Islam based on the ternary statistical language model of Google Web 1T [13] data sets, using the longest common subsequence matching algorithm to check and correct the real-word errors in English and obtains 89% errors checking recall and 76.3% errors correction recall [14]. However, Berlinsky's study showed that data sparsity may make the model with larger N-Gram less effective than the model with smaller N-Gram [15]. So there is a need to balance the N-Gram information and data sparsity, so Islam used multivariate fusion compensation to combine a five-Gram statistical model to a binary statistical model for real-word errors checking and correction [16]. The machine learning method uses a predefined set of real-word confusions to learn the contextual features of the words and comparing the words in the confusion set for real-word errors correction, the method needs to rely on the feature training set, and the learning method of annotation is more cumbersome and not well generalized and scalable [17]. Kochmar and Briscoe [18] have applied the same semantic combination model to distinguish correct and incorrect ANs, and their results support the hypothesis that there are distinguishable differences between the composite vectors of correct and incorrect ANs, but they don't address the question of how to integrate these semantic models into an errors detection system. Hirst [19] calculates the semantic distance between the target word and the contextual word by WordNet semantic lexicon, with the closer distance being the correct word and that are farther away are real-word errors, and the highest recall rate of the experiment is near 50%, and the highest precision is near 20%. Several reasons analyzed by Hirst, including the limitations of the WordNet dictionary, the efficiency of search-

ing words, the difference between similarity and semantic relatedness, the threshold value setting of relatedness, proper nouns, the limitation of method evaluation and so on are the factors that restrict the experiment effect. Although the experimental effect of Hirst's research is unsatisfactory, has positive implications for the subsequent research on checking and correcting real-word errors based on semantic information to a certain extent.

Table 1. The comparison of existing word errors correction methods

	Methodology	Concept/ Feature
Correcting non-word errors	Minimum Edit Distance	Correcting candidates are determined by calculating the minimum edit distance between misspelled strings and words in the dictionary.
	Similar Skeleton Key	By constructing the skeleton key dictionary, the correct words with the same skeleton key in the dictionary are used as the correction suggestions.
	Misspelling Statistical Dictionary	Collecting misspelled English words from real texts and give the correct spelling, but the unambiguities and comprehensiveness of misspelled dictionaries is difficult.
	N-Gram Statistical Analysis	Through the statistics of English texts, getting the transfer probability matrix between words, and transfer probability is greater than a given threshold value are suggested for error correction.
Correcting real-word errors	Machine Learning	Relying on the feature training set, and the learning method of annotation is more complicated and has poor generalization and expansibility.
	Semantic Information	Real-word errors checking is based on contextual semantic relations, assuming that the right word has a strong semantic connection to its context, while the real-word errors does not have such a semantic association.
	N-Gram Statistical Language Model	Correcting errors approach relies on huge N-Gram statistical model, capturing longer semantic relations is difficult.

3 Methodology

The English word errors correction model classifies word errors into non-word errors and real-word errors. The checking and correction of isolated non-word errors relies mainly on the word form and the pronunciation of its representation, while the checking and correction of real-word errors relies mainly on meaning of the word in its context. The overall structure of the word errors correction model in this paper is shown in Fig. 1.

For non-word errors that do not exist in the dictionary, compiling a dictionary and constructing a dictionary tree to check the errors. Non-word errors correction model based on edit distance and phoneme matching is built to provide suggestions in the form of a candidate list for correction of non-word errors. For real-word errors correction model, an errors checking method based on the confusion set of real-words, and the binary statistical model is used to supplement the suggestions of candidate real-word errors. Combined with the GloVe to give corrective suggestions for real-word errors in comparison with contextual word meanings. The methods of the two modules are described in detail below.

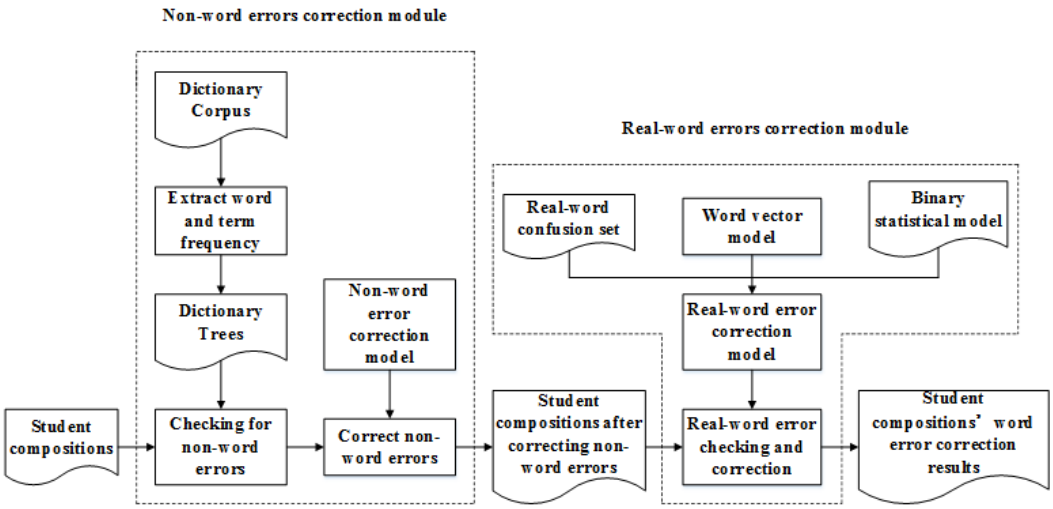

Fig. 1. Overall structure of word errors correction model

3.1 Non-word Errors Correction Methods

3.1.1 Constructing Dictionary

Most non-word errors are caused by single incorrect operation such as deletion, substitution and insertion. To address the non-word errors in students' English compositions, this paper uses a dictionary search method to solve the problem of word checking, where a word in a composition is considered to be anon-word errors when it does not exist in the dictionary. In this paper, English words are divided into ordinary words and special words. The special words include words in abbreviated form with single inverted commas, words with hyphens, date abbreviations and proper nouns. Because the number of special words is less than that of common words, a special word dictionary is compiled separately to check the errors of such words in the way of traversal, and the common words that account for the majority of words are searched in the way of dictionary tree. In this paper, the English texts from the data such as university English 4-6 CET compositions models and English textbooks for university students are selected as the corpus for the construction of the dictionary, in order to meet the requirements of word correction for university students' English compositions.

Next, the ordinary word dictionary is constructed into the form of a dictionary tree. A dictionary tree is a tree-shaped storage structure, where each node except the root node has one and only one character, and all child nodes of each node have different characters, so that invalid string comparisons can be avoided to the greatest extent possible and queries are efficient. The dictionary is mapped into a dictionary tree to improve the efficiency of non-word errors checking for ordinary word forms. When building a dictionary tree, the first step is to clarify its internal structure. A node within a dictionary tree must contain two basic properties, one being the value to be stored and the other being the child nodes it contains. For example, if the word "wonderful" has a node path constructed in the dictionary tree, then the node path for the string "wonderfu" also exists in the dictionary tree, this requires a flag to indicate whether the current node and path is a complete representation of a word, i.e. whether it is a final node.

In order to preserve the word frequency of a word, a node's word frequency attribute needs to be added. When the node mark is the end of a certain word, the word frequency attribute is the word frequency of the word; when it is not the end node, the word frequency attribute of the node is 0. The final dictionary tree thus contains four attributes: the value of the node, the child node, whether the node is a final node and the word frequency. Once the dictionary tree is built, it can be used to check for non-word errors. The steps of looking up words using the dictionary tree are very similar to the way of establishing the path of word nodes.

3.1.2 Constructing Dictionary

The word errors correction module performs non-word errors correction, it first determines whether a boundary errors has occurred in a word. A boundary error is a word error caused by the wrong input of space in the middle

of a correctly spelled word or a missing space in the middle of two correctly spelled words. The former is type I boundary errors and the latter is type II boundary errors. We use the method of minimum edit distance to correct non-word errors based on character distance, and find the word with the minimum edit distance from the non-word errors in the dictionary as the suggested word.

Take the word “academic” as an example, “acdemic” is a deletion error, “academac” is a replacement error, and “accademic” is an insertion error, and “acadmeic” is an interchange error. In this paper, the distance calculation method treats the distance cost of deletion, replacement and insertion of characters in words as 1, and the distance cost of character interchange as 2, so as to encompass the four types of English word input errors, and to obtain suggestions for the correction of non-word errors caused by input errors from the compiled English dictionary.

The formula for $d_{i,j}$ is shown in the following equation (1). $d_{i,j}$ is recorded as the value in the cell of the row i and colu mn j, and the initial value $d_{1,1}$ is set to 0. a_i is the ith character in string a, b_j is the jth character in string b. By setting the threshold, the correctly spelled words in the dictionary that do not exceed the minimum edit distance from the non-word errors word to the threshold value, which is saved in the candidate list of correction suggestions for non-word errors.

$$\mathrm{d}_{i,j}=\begin{cases} d_{i-1,j-1} & for \quad a_i = b_j \\ \min\begin{cases} d_{i,j-1}+INS(a_i) \\ d_{i-1,j-1}+SUB(a_i \cdot b_j) & for \quad a_i \neq b_j \\ d_{i-1,j}+DEL(b_j) \end{cases} \end{cases} . \tag{1}$$

3.1.3 IDM Algorithm

The editing distance between the cognitively confused incorrect words and the correct words is not small, so a phoneme-based matching method is needed to correct the word errors. This paper uses IDM algorithm (Improving Double Metaphone) to implement the non-word input errors correction method.

The Double Metaphone algorithm maps words into the form of consonant keys to match the correct words with similar pronunciation in the dictionary, suggestions are provided for correcting non-word ambiguous matching of consonant phonemes. But the Double Metaphone original rule set has no rules to code vowel phonemes, we improve the original Double Metaphone algorithm by adding the vowel mapping set, and get total of 20 vowel phonemes and their specific character combinations, since the vowel phoneme “[æ]" contains only one letter "a", there is no problem of vowel spelling confusion, after eliminating it. When a non-word error is detected, the vowel character combinations of the word are mapped to the corresponding vowel symbols in turn, and each matching mapping only matches the current position to the corresponding key value, providing suggestions for vowel character replacement of non-word errors through vowel symbol and mapping matching. The combination of vowel characters in a word may match multiple strings in the same list, in this case the replacement will be performed sequentially, all mapped characters will be replaced the words where the non-word errors occurs, all the replaced and repeated words will be put into the constructed dictionary tree for searching, and the correctly spelled words will be obtained and added to the suggested candidate list. The vowel phoneme mapping set is used as follows.

(1) Detecting the character combinations corresponding to the vowel phonemes contained in the non-word errors words.

(2) Replacing other character combinations of that vowel, one by one if there are multiple vowels.

(3) Putting all the obtained replacement words into the previously constructed dictionary tree to find the correctly spelled words.

(4) Sorting the candidate words by edit distance.

(5) If the list of candidate words is too long, it is handled by discarding the trailing part.

(6) Generating phoneme matching non-word errors correction suggestions by combining the candidate words of IDM algorithm.

Based on the single error principle, IDM algorithm consonant matching is performed separately from vowel phoneme mapping set matching. After getting the suggestions given by the phoneme matching-based non-word errors correction method, the word frequencies of the candidate suggested words are extracted from the dictionary tree, and all the candidate suggested words are sorted by editing distance. The suggested words with the same edit

distance are then sorted by word frequency, and only a certain number of words will be kept as output suggestions when the suggestion list is too long to give the non-word errors correction suggestions for students' English compositions.

3.2 Real-word Errors Correction Methods

3.2.1 Binary Statistical Model

In this paper, we use the open source real-word confusion sets provided by After the Deadline to predetermine the real-word errors in students' English compositions based on real-word confusion sets, which are collections of words that are often confused and used, mostly homophones and anagrams, but most of the real-word confusion sets contain only two words. If the candidate words can be added, we can achieve better results in correcting real-word errors. The model presupposes the real-word errors as confusion, and puts the other words in the same confusion set into the candidate list with the words obtained by matching the binary statistical model, and calculates the contextual semantic relevance together with the target words. By combining the binary statistical model, we provide for extending candidate words for real-word errors, choosing Norvig's collated binary model combined with LCS algorithm to obtain suggested words outside the real-word confusion set.

This model takes out the previous word and the next word of the target word predetermined as real-word errors to form two binary phrases respectively, and the conditional equation of the matching binary statistical model is shown in equation (2) .

$$\mathrm{L}(t)-2 \le LCS(t,m) \le L(t) and L(t)-2 \le L(m) \le LCS(t,m)+2 \ . \quad \textbf{(2)}$$

In equation (2), t is the target word, m is the matching word similar to the target word in the binary statistical model, L(t) is the length of the target word, L(m) is the length of the matching word, and LCS(t, m) is the length of the longest common subsequence so as to obtain the appropriate binary phrase and its matching candidate words in the binary statistical model. of the matching word and the target word, After getting the matching words in the binary statistical model, the LCS(t, m) of the matching words and the target words are regularized for the convenience of weight calculation, which is noted as NLCS(t, m), and the regularization formula is shown in the following equation (3).

$$\mathrm{NLCS}(t,m)=\frac{2\times LCS(t,m)}{L(t)+L(m)} \ . \quad \textbf{(3)}$$

Also, the frequencies of the matched words are regularized. The frequencies of n matching words T_1, T_2, T_3...T_n are f_1, f_2, f_3...f_n, and the frequency of matching word m is denoted as f_m, and the regularized frequency is denoted as NF(m), m ∈ [1, n]. The frequency regularization is calculated as shown in equation (4) .

$$\mathrm{NF(m)}=\frac{\mathrm{f_m}}{\max(f_1,f_2,f_3,L,f_m,L,f_n)} \ . \quad \textbf{(4)}$$

To balance the influence of the length and frequency of matched words on word weights, the adjustment factor is set to λ, λ ∈ (0, 1), and the larger the adjustment factor, the greater the influence of word frequency on the weight of matched words. The smaller the adjustment factor, the greater the influence of the length of the longest common subsequence between the matched word and the target word on the weight of the matched word. The weight W_m of matching word m is calculated as shown in equation (5).

$$W_m = (1\text{-}\lambda) \times NLCS(t, m) + \lambda \times NF(m) \ . \quad \textbf{(5)}$$

Since the matching rule of the longest common subsequence is set earlier, it is necessary to reduce the influence of subsequence length on the weight of matching words and increase the proportion of word frequency in the weight, and λ=0.8 is set here. The matching words obtained from the binary statistical model are sorted by the weight size, and the first two are taken out as the candidate suggested words, and then put together with the target words that are preset as real-word errors and the words in their real-word confusion set. After removing the duplicate words, the GloVe word vector model trained in the following section is used to calculate the semantic

relevance of each word to the context of the target word in the English compositions, and to compare whether the word is a real-word error and the suggested word to correct the real-word errors.

3.2.2 Word Embedding Model

The word vectors generated by the GloVe are used to calculate the semantic relevance of words in context, and the word vectors are used to represent the word meanings, and the cosine similarity is used to calculate the similarity between the word vectors. They are filtered using the previously constructed dictionary tree to obtain the word vector model used to calculate the contextual semantic relevance of target words or candidate words.

Setting the word vector of the target word or candidate word as $X=(x_1, x_2, x_3,..., x_n)$ and n is the dimension, and the word vector of a certain context word as $Y=(y_1, y_2, y_3,..., y_n)$, the formula to calculate the semantic correlation R between two words is as follows (6) is shown.

$$\mathrm{R}(X,Y)=\frac{\sum_{i=1}^{n} x_i \times y_i}{\sqrt{\sum_{i=1}^{n}(x_i)^2}\times\sqrt{\sum_{i=1}^{n}(y_i)^2}} \quad . \tag{6}$$

Before selecting the target word context, the target word and the candidate word are lexically labeled. When the two words are same, the context window size is set to 3, i.e., it contains total of 7 words of the target word, and three words on the left and right sides of the target word are extracted as the reference words for calculating the semantic relevance of the context; When the word nature is different, setting the context window size to 2, i.e., the target word contains 5 words, and extract two words on the left and right of the target word as the reference words. The target word itself is used to calculate the score of contextual semantic relevance with a context window of 3.

The size of the context window is ContextWindow, R_i denotes the semantic relevance score of the target word or candidate word with the ith word in the context window, $i \in [0, 2\times\mathrm{ContextWindow}]$, and the formula for calculating the ContextScore of the context relevance score of the target word or candidate word is shown in the following equation (7).

$$\mathrm{Con}textScore=\frac{R_1+\cdots+R_{ContextWindow}+R_{ContextWindow+1}+\cdots+R_{2\times ContextWindow}}{2\times ContextWindow} \quad . \tag{7}$$

When there is a non-word error in the context window of the selected target word, using the correction suggestions from the previous non-word correction module and replacing them with context, in order to better calculate the contextual relevance score of the target word or candidate word. Let the first n correction suggestions $w_1, w_2,..., w_n$ of this non-word errors word be selected as replacement, setting different weight coefficients for these n words, corresponding to the first correction suggestion word to the nth suggestion word as $\beta_1, \beta_2, ..., \beta_n$, to replace the number of correct suggested words and assign weight coefficients in reverse order. For example, if n=3 suggested words are replaced to the non-word errors position in the context window, then $\beta_1=3$, $\beta_2=2$, $\beta_3=1$. ContextScore(w_k) is the result of calculating the score of equation (6) for the replacement of the k-th word of the non-word errors correction suggestion into the context window, $k \in [1,n]$. The higher the ranking of the correction suggestion words, the higher the weight factor. In general, there is at most one non-word error word in the context window, when there is more than one non-word errors, except the first non-word error word, only the first non-word error correction suggestion is replaced. The contextual relevance score for the presence of non-word errors words in the context window is calculated as shown in equation (8).

$$\mathrm{Con}textScore_{\mathrm{Sub}stitute}=\frac{\sum_{k=1}^{n}\beta_k\times ContextScore(w_k)}{\sum_{k=1}^{n}\beta_k} \quad . \tag{8}$$

The algorithm pseudo-code for calculating contextual relevance score of candidate words is shown in algorithm 1.

Algorithm 1. Pseudo-code for calculating the semantic relevance score of word context

```
Input: Candidate, List, Vector
Output: ContextScore
For word In Candidate Do
  If POS(string)=pos
Then contextWindow←2
Else contextWindow←3
Extract(List, contextWindow)
If List.Contain(non-word)
Then CalculateContextScoreWithSubstitute(string, List, Vector)
Else CalculateContextScore(string, List, Vector)
End
Return ContextScore
```

4 Experiment

The experimental data for the English composition word errors correction module mainly include: Cet-4 and CET-6 Composition samples and College Students' English textbooks for constructing English dictionaries; Based on the Chinese Learners English Corpus of 10,000 English texts by CET-4 and CET-6 non-English majors [20], to evaluate the correction effect of non-word errors and content word errors. Conducting a comparison experiment between model scoring and manual scoring, and English compositions in different knowledge fields, difficulty levels and topic types were selected as experimental data sets. Team English teachers were invited to evaluate the correlation between the model and manual scoring, so as to verify whether the model can replace the general manual scoring model in English subjective questions.

4.1 Non-word Errors Correction Experiment

After processing the datasets with the non-word errors correction method of the word correction module of this model, relatively strict non-word errors correction criteria are selected, and the top three non-word errors correction suggestions are used as the basis for discrimination. If one of the first three suggested words is correct, the non-word errors is considered to be successfully corrected.

The statistical results of non-word errors in different text number during the experiment are shown in Table 2.The above statistical results of non-word errors checking and correction were compiled, and calculate the accuracy rate, recall rate and F_1 value of non-word errors correction in this module at different text number, the statistics were presented in the form of a bar chart as shown in Fig. 2.

Table 2. Statistical results of non-word errors in different text number

Text number	Non-word errors number		
	Marking	Finding	Correcting
500	1283	1224	1043
1000	2147	2038	1746
3000	5932	5636	4855
5000	10653	10126	8734
10000	21370	20312	17570

In order to verify the ability of the model's non-word errors correction method to the suggestion words in the candidate suggestion list, the experiment uses a comparison method with the widely used error correction system.

In this paper, 2,000 students' English compositions were selected as a comparative experimental data set, which contained 6550 non-word errors markers. In the experimental environment, the program interface provided by JaSpell and Hunspell was used to check and correct the selected experimental data set for non-word errors, the suggested word orderings for correcting non-word errors are classified into Top1, Top3, Top6, Top9, and the experimental data are collated to obtain the comparison results of the accuracy rate of correcting non-word errors for the four suggested word orderings is shown in Fig. 3.

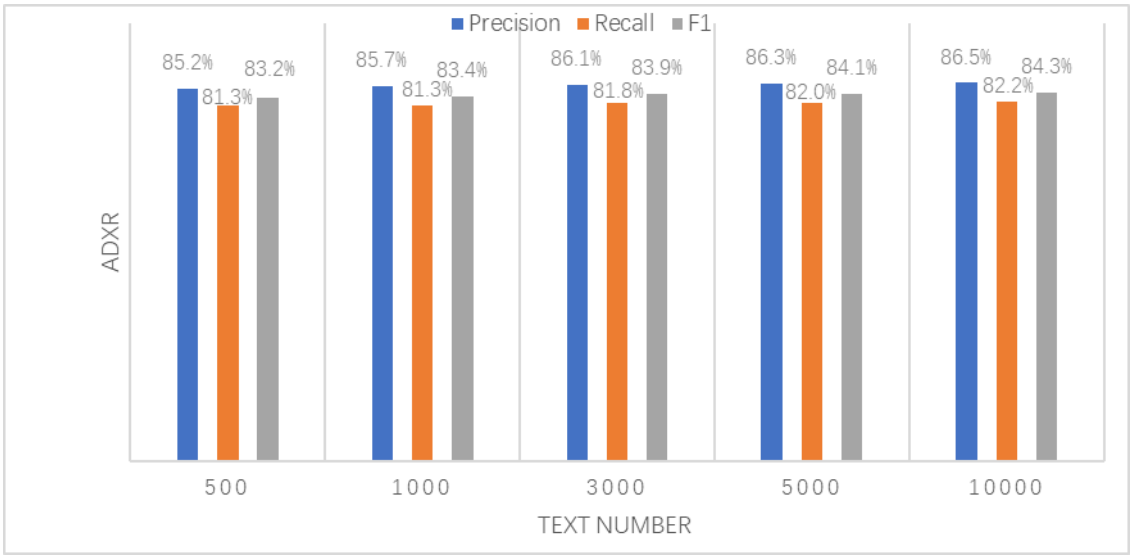

Fig. 2. Evaluation values of non-word errors correction for different text number

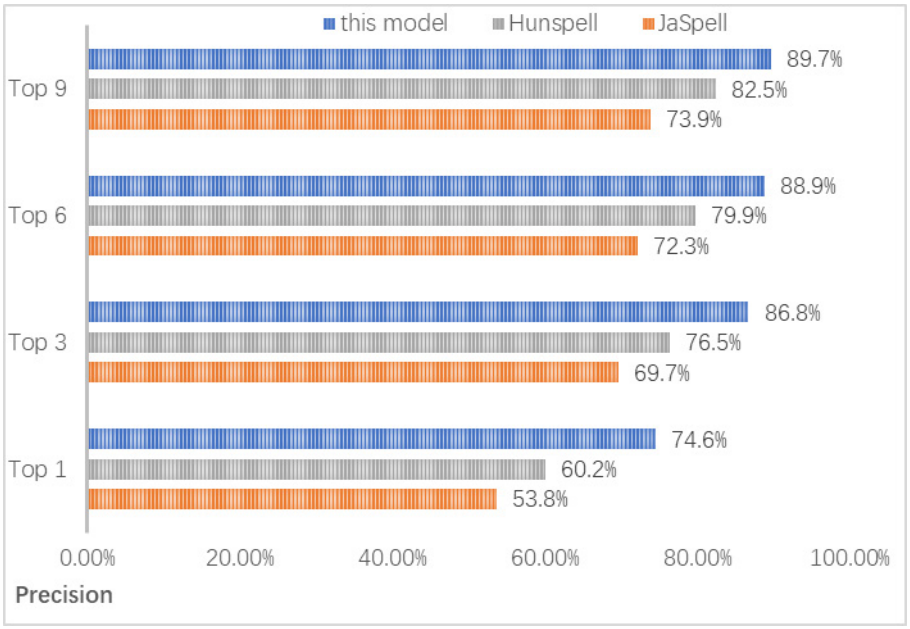

Fig. 3. Comparison of the precision of correcting non-word errors for the proposed word ordering

Through the above experiments, it can be seen that the non-word errors checking and correction method of this model can obtain good results and has higher errors correction accuracy compared with other correction systems. It can achieve 86.5% accuracy of non-word errors correction, 82.2% recall rate and 84.3% F_1 value when relatively strict non-word errors correction criteria are selected. The experiments show that the model's non-word errors correction method can better check and correct the non-word errors in Chinese students' English short texts.

4.2 Real-word Errors Correction Experiment

Using the real-word errors correction method in the experimental data set, determining whether they are real-word errors in the results, and gives correction suggestions if they are. In the statistical 10,000 student English composition corpus, there are 10961 marked real-word errors in total, the real-word errors correction method of the word correction module of this model detects 8760 real-word errors, among which 6820 real-word errors are correctly corrected, and the accuracy rate of real-word errors correction is 77.9%, the recall rate is 62.2%, F_1 value is 69.2%.The statistical results of real-word errors in different text number during the experiment are shown in Table 3.

Table 3. Statistical results of real-word errors in different text number

Text number	Real-word errors number		
	Marking	Finding	Correcting
500	731	589	457
1000	1350	1089	840
3000	3227	2595	2014
5000	5183	4153	3232
10000	10961	8769	6820

The above statistical results of real-word correction were compiled, and the accuracy rate, recall rate and F1 value of real-word correction in the word correction module of this model at different text number were calculated, and the statistics were presented in the form of bar graphs as shown in Fig. 4.

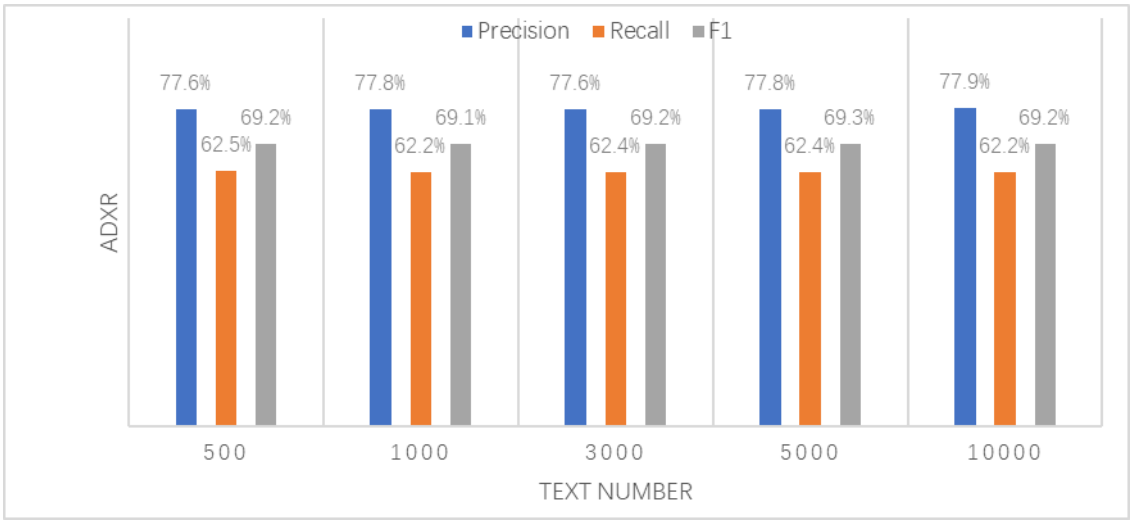

Fig. 4. Evaluation values of real-word errors correction for different text number

Real-word errors correction has been a difficult problem in the field of word errors correction research. The real-word errors correction approach based on real-word confusion sets usually has complex contextual feature labeling and training, so tens of real-word confusion sets are often selected for experimental evaluation, but this approach is poorly scalable, and the recall of errors correction is basically determined by the number of real-word confusion sets for all real-word errors in the text, so there are some practical application limitations in practical applications. This model adopts an easy-to-use and clear real-word errors correction method, which can apply a larger real-word confusion set for real-word errors correction, has better scalability, can increase or decrease the number of real-word confusion sets as the application area changes without the operation of labeling and training, and takes into account the accuracy and recall rate of real-word errors correction.

The experiments show that this model can achieve 77.9% correction accuracy of real-word errors correction,

and the number of nearly one thousand real-word confusion sets ensures the recall of correction to a certain extent, which has good practical application value and can be used to correct the real-word errors in Chinese students' English compositions.

4.3 Comparison with Manual Scoring

In this scoring setting, the word score of English subjective questions is set to a full score of 30 points, and the experimental corpus was scored by the word correction module of this model and manually scored respectively, and the comparison of the experimental results is shown in Fig. 5.

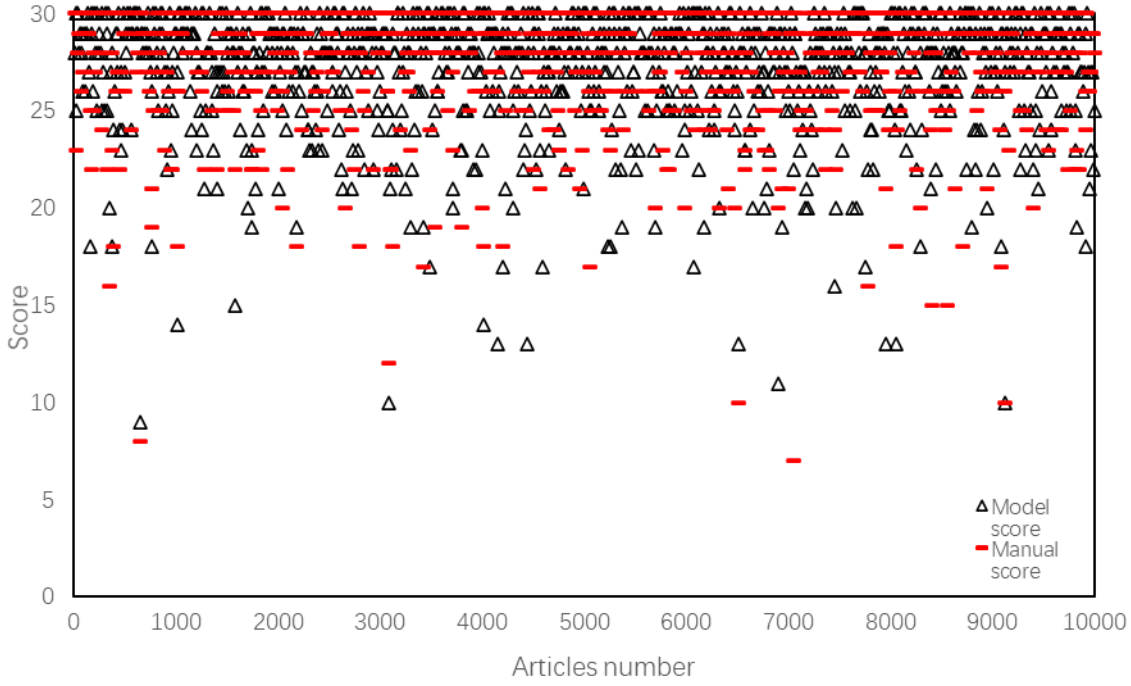

Fig. 5. Comparison of word correction model scoring and manual scoring

From the scatter plot of the experimental comparison, it can be seen that, with students' English composition as the experimental target, the automatic scoring results of word correction module of the model and the manual scoring results are relatively similar, and the average score of the model's scoring is 27.65 and the average score of the manual scoring is 26.90. The average errors of the two scoring is 0.93, and the Pearson correlation coefficient is 0.81. The experiments show that the word correction module of the model can be used to check and correct the word errors in Chinese students' English compositions, which has certain practical application value.

5 Conclusion

In this paper, designing a non-word errors correction method combining character distance and phoneme matching, a real-word errors correction method based on the real-word confusion set and combining binary statistical model and Glove vector model. The errors correction accuracy of the non-word errors correction model was higher than that of Japell and Hunspell, which reached 86.5% accuracy; the real-word errors correction model achieves 77.9% accuracy of correction and guarantees a certain degree of recall. However, the correct method of real-word errors correction relies too much on the confounding set, and the semantic disambiguation ability depends on the quality of the word vector model. Next, replacing a larger confounding set of real-word, or selecting a sliding context window to perform semantic correlation analysis.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62066009).

References

[1] K. Knill, M. Gales, P. Manakul, A. Caines, Automatic Grammatical Error Detection of Non-native Spoken Learner English, in: Proc. ICASSP, IEEE, 2019.
[2] V.M. Christanti, S. Naga. Dali, Fast and accurate spelling correction using trie and Damerau-levenshtein distance bigram, Telkomnika 16(2)(2018) 827-833.
[3] S.M. Dashti, Real-word error correction with trigrams: correcting multiple errors in a sentence, Language Resources and Evaluation 52(2)(2018) 485-502.
[4] M. Octaviano, A. Borra, A spell checker for a low-resourced and morphologically rich language, in: Proc. TENCON 2017 - 2017 IEEE Region 10 Conference, 2017.
[5] Y. Hong, X. Yu, N. He, N. Liu, J. Liu, FASPell: A Fast, Adaptable, Simple, Powerful Chinese Spell Checker Based On DAE-Decoder Paradigm, in: Proc. of the 5th Workshop on Noisy User-generated Text, 2019.
[6] L. Nemeth, Hunspell, http://hunspell.sourceforge.net, 2018.
[7] L. Al-Hussaini, Experience: Insights into the Benchmarking Data of Hunspell and Aspell Spell Checkers, Journal of Data and Information Quality (JDIQ) 8(3-4)(2017) 1-10.
[8] J. Pennington, R. Socher, C.D. Manning, Glove: Global vectors for word representation, in: Proc. of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014.
[9] S. Singh, HINDIA: a deep-learning-based model for spell-checking of Hindi language, Neural Computing and Applications 33(8)(2021) 3825-3840.
[10]A. Pal, S. Mallick, A.R. Pal, Detection and Automatic Correction of Bengali Misspelled Words using N-Gram Mode, in: Proc. 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), IEEE, 2021.
[11]Y. Qin, Y. Qian, A. Loukina, P. Lange, A. Misra, K. Evanini, T. Lee, Automatic Detection of Word-Level Reading Errors in Non-native English Speech Based on ASR Output, in: Proc. 2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP), IEEE, 2021.
[12]E. Kochmar, E. Shutova, Cross-Lingual Lexico-Semantic Transfer in Language Learning, Association for Computational Linguistics, 2016.
[13]M. Flor, A fast and flexible architecture for very large word n-gram datasets, Natural Language Engineering 19(1)(2013) 61.
[14]A. Islam, D. Inkpen. Real-Word Spelling Correction using Google Web 1T 3-grams, in: Proc. of the 2009 Conference on Empirical Methods in Natural Language Processing, 2009.
[15]S. Singh, Handling Real-word Error of Hindi Language using N-gram and Confusion Set, in: Proc. 2019 Amity International Conference on Artificial Intelligence (AICAI), IEEE, 2019.
[16]A. Islam, D. Inkpen, Real-word spelling correction using Google Web 1T n-gram with backoff, in: Proc. 2009 International Conference on Natural Language Processing and Knowledge Engineering, IEEE, 2009.
[17]A.M. Azmi, M.N. Almutery, H.A. Aboalsamh, Real-word errorS in Arabic texts: A better algorithm for detection and correction, IEEE/ACM Transactions on Audio, Speech, and Language Processing 27(8)(2019) 1308-1320.
[18]M. Kaneko, Y. Sakaizawa, M. Komachi, Grammatical error detection using error-and grammaticality-specific word embeddings, in: Proc. of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2017.
[19]R. Ferreira, R.D. Lins, S.J. Simske, F. Freitas, M. Riss, Assessing sentence similarity through lexical, syntactic and semantic analysis, Computer Speech & Language 39(2016) 1-28.
[20]M. Liu, Design of Intelligent English Writing Self-evaluation Auxiliary System, Informatica 43(2)(2019).