A Sequence to Sequence Learning for Chinese Grammatical Error Correction

Hongkai Ren[1,2], Liner Yang[1,2](✉), and Endong Xun[1,2]

[1] Beijing Advanced Innovation Center for Language Resources, Beijing, China
[2] School of Information Science, Beijing Language and Culture University, Beijing, China
renhongkai27@gmail.com, yangtianlin08@gmail.com, edxun@126.com

Abstract. Grammatical Error Correction (GEC) is an important task in natural language processing. In this paper, we introduce our system on NLPCC 2018 Shared Task 2 Grammatical Error Correction. The task is to detect and correct grammatical errors that occurred in Chinese essays written by non-native speakers of Mandarin Chinese. Our system is mainly based on the convolutional sequence-to-sequence model. We regard GEC as a translation task from the language of "bad" Chinese to the language of "good" Chinese. We describe the building process of the model in details. On the test data of NLPCC 2018 Shared Task 2, our system achieves the best precision score, and the $F_{0.5}$ score is 29.02. Our final results ranked third among the participants.

Keywords: Grammatical Error Correction
Convolutional Sequence to Sequence Model
Neural machine translation

1 Introduction

The rapid development in China attracts people from all over the world to learn Chinese. Chinese is a historically influential and versatile language. Chinese is unique in many aspects as opposed to English and other languages. One of the distinction worth mentioning is its lack of verb conjugations and plural suffixes. Besides, the sentence expression is very flexible, which means that the rearrangement of word order in various ways may not impact on the sentence meaning. While handling grammatical complexity comes very naturally to native Chinese speaker, to be proficient and competent is very challenging to CSL (Chinese as Second Language) learners. Therefore, it is practical to develop a system automatically correcting grammatical errors, which is the goal of the NLPCC 2018 Shared Task 2.

English grammar correction has been studied for many years with great progress. In particular, after Ng et al. [13] organize the CONLL-2013 shared task, a large number of methods based on statistics and neural networks emerge, which greatly promote the study of English grammar error correction. Chollampatt et al. [1] propose the phrase-based statistical machine translation (SMT) approach, in which GEC is firstly treated as a translation task. By training the model to "translate" the "bad" English into the "good" English, they carry out very promising results. Following the previous work, several neural encoder-decoder approaches have been put forward for this task. Chollampatt et al. [2] firstly employ a convolutional encoder-decoder model that achieved good performance for GEC. Among those, Junczys-Dowmunt et al. [9] demonstrate parallels between neural GEC and low-resource neural MT. They successfully adopt several methods from low-resource MT to neural GEC, and achieve the state-of-the-art results on this task.

While English Grammatical Error Correction is being intensively studied for years, the same task on Chinese is poorly focused until very recently. In 2014, Yu et al. [10] organize a Shared Task on Grammatical Error Diagnosis (GED) for Learning Chinese as a Foreign Language (CFL). The goal of this shared task is to develop computer-assisting tools for GED of several kinds (i.e., redundant word, missing word, word disorder, and word selection). The task had led researchers to focus on Chinese grammar errors correction in computational linguistics. Until 2017, Rao et al. [14] organize the IJCNLP 2017 Shared Tasks on CGED, where the task still solely concentrates on the detection of the grammatical errors rather than the automatic generation of corrections. The NLPCC 2018 Task 2 gives NLP researchers an opportunity to develop the Chinese grammatical error correction system.

This paper is organized as follows: Sect. 2 describes the GEC shared task. Section 3 illustrates the details of our structure. In Sect. 4, we present our experiment in details, including the data preprocessing and results. In Sect. 5, we introduce some related work both in English and in Chinese. Last but not least, the conclusion and a prospect of future work are given in Sect. 6.

2 Grammatical Error Correction

With the expanding influence of China, learning Mandarin Chinese has grown in popularity around the world. Even though the study of second language learning has started many years ago, the research of CSL still has a long way to go.

The goal of the NLPCC 2018 Shared Task 2 is to evaluate algorithms and systems for the automatic detection and correction of grammatical errors from second language learners of Chinese. Given a Chinese sentence, a GEC system is expected to correct four types of grammatical errors, including redundant words (R), missing words (M), bad word selection (S) and disorder words (W).

A grammatical error correction system is evaluated by how well its proposed corrections or edits match the gold-standard edits. A sentence is first segmented before evaluation is carried out on a set of sentences. The metrics measured at the testing stage are: Precision, Recall and $F_{0.5}$. Let g_i is the set of gold-standard edits for sentence, and e_i is the set of system edits for sentence. The measurements are defined as follows:

$$P = \frac{\sum_{i=1}^{n} |e_i \cap g_i|}{\sum_{i=1}^{n} |e_i|}, \tag{1}$$

$$R = \frac{\sum_{i=1}^{n} |e_i \cup g_i|}{\sum_{i=1}^{n} |g_i|}, \tag{2}$$

$$F_{0.5} = \frac{(1+0.5^2) \times R \times P}{R + 0.5^2 \times P}, \tag{3}$$

where the intersection between e_i and g_i is defined as:

$$e_i \cup g_i = \{e \in e_i | \exists g \in g_i \ , match(e, g)\}. \tag{4}$$

We choose $F_{0.5}$ which emphasizes precision twice as much as recall as our F-measure for that when a grammar checker is put into actual use, the accuracy of its corrections is profoundly valued in order to gain users' acceptance. Negligence in offering a correction is not as bad as giving a wrong one. The NLPCC 2018 Shared Task 2 use the MaxMatch (M^2) scorer[1] [4] as the official scorer. The M^2 scorer efficiently searches for a set of system edits that maximally matches the set of gold-standard edits specified by an annotator.

3 Methodology

Sequence-to-sequence model has been proven to be powerful in many tasks such as machine translation [12], speech recognition [3] and text summarization [15]. Our model for the task of GEC, inspired by the work of Gehring et al. [6], is based on a fully convolutional encoder-decoder architecture with multiple layers of convolutions and attention mechanisms. Most grammatical errors are often localized and dependent more heavily on the nearby words. Therefore, we take advantage of the convolutional neural networks (CNNs), as it can capture local context more effectively than RNNs by performing on smaller windows over the word sequences. Wider contexts and interactions between distant words can also be captured by a multilayer hierarchical structure of convolutions. Moreover, an attention mechanism that assigns weights over the source words based on their relevance is used when predicting the target word. One benefit of our model is that only a fixed number of nonlinear operations are implemented on the input disregarding its length, whereas when using RNNs, the number of nonlinear operations is proportional to the length of the input, diminishing the effects of distant words. In the following section, we will describe our model in details.

3.1 Convolutional Sequence to Sequence Model

We embed input source sentence S given as a sequence of m source words $s_1, \cdots, s_m$ then lookup embedding vector from embedding matrix for each word

[1] http://www.comp.nus.edu.sg/~nlp/software.html.

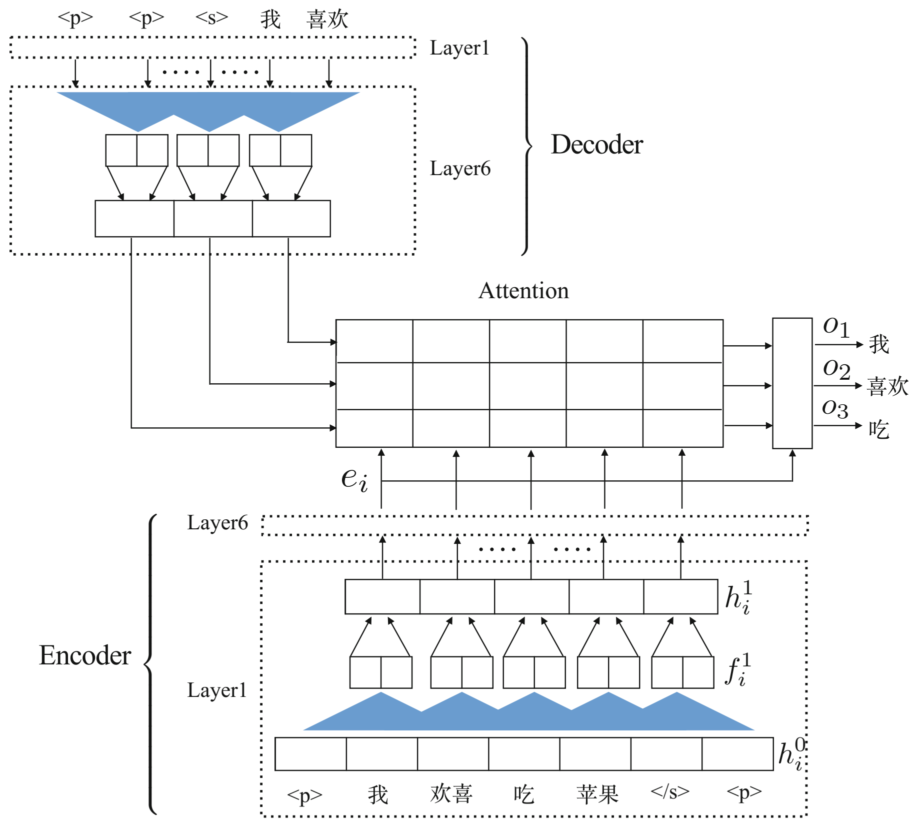

Fig. 1. The architectures of convolutional sequence-to-sequence model.

s_i as $w_{s_i} \in \mathbb{R}^d$. We also equip our model with a sense of order by embedding the absolute position of input elements $p = (p_1, \cdots, p_m)$ where $p_j \in \mathbb{R}^d$. Both are combined to obtain input sentence representations $s = (w_1 + p_1, \cdots, w_m + p_m)$. We proceed similarly for output elements that were already generated by the decoder network to yield output element representations that are being fed back into the decoder network $g = (g_1, \cdots, g_n)$. Position embeddings are useful in our architecture since they give our model a sense of which portion of the sequence in the input or output it is currently dealing with.

In this section, we will describe our model in details. The encoder and decoder are made up of L layers each, share a simple block structure that computes intermediate states based on a fixed number of input elements. Each block contains a one dimensional convolution and a non-linearity. The goal of this framework is to estimate the conditional probability $p(y_{i+1}|y_1, \cdots, y_i, S)$, where S is an input sentence and $\{y_1, y_2, \cdots, y_m\}$ is the corresponding output sequence. The architecture of the network is indicated in Fig. 1.

Encoder. Pass the source token embeddings, $s_1, \cdots, s_m$, over the linear layer to get the input vectors of the first encoding layer, $h_1^0, \cdots, h_m^0$, where $h_i^0 \in \mathbb{R}^d$ and h is the input and output dimension of all encoder and decoder layers. In the first encoder layer, convolution kernel is parameterized as $W \in \mathbb{R}^{(2d \times kd)}$, $b_w \in \mathbb{R}^{2d}$ and takes as input $X \in \mathbb{R}^{k \times d}$ which is a concatenation of k input elements embedded in d dimensions and maps them to a single output element $Y \in \mathbb{R}^{2d}$ that has twice the dimensionality of the input elements. Paddings (denoted by in Fig. 1) are added at the beginning and end of the source sentence to retain the same number of output vectors as the source tokens after the convolution operations.

$$Y = [A \quad B] \in \mathbb{R}^{2d} \tag{5}$$

This is followed by a non-linearity using gated linear units (GLU) [5]:

$$GLU(Y) = A \otimes \sigma(B) \tag{6}$$

where $A, B \in \mathbb{R}^d$ are the inputs to the non-linearity, $\otimes$ and σ represent element-wise multiplication and sigmoid activation functions, respectively. To enable deep convolutional networks, we add residual connections from the input vectors of each encoder layer to the output of the layer. The output vectors of the 1^{th} encoder layer are given by,

$$h_i^l = GLU(Y) + h_i^{l-1} \qquad i = 1, \cdots, m \tag{7}$$

Each output vector of the final encoder layer, $h_i^L \in \mathbb{R}^h$, is linearly mapped to get the encoder output vector, $e_i \in \mathbb{R}^d$, using weights $W_e \in \mathbb{R}^{d \times h}$ and biases $b_e \in \mathbb{R}^d$:

$$e_i = W_e h_i^L + b_e \qquad i = 1, \cdots, m \tag{8}$$

Decoder. Each decoder layer has its own multi-step attention. To compute the attention, we combine the current decoder state $y_n^l \in \mathbb{R}^h$ with an embedding of the previous target element t_{n-1}:

$$z_n^l = W_z y_n^l + b_z + t_{n-1} \qquad W_z \in \mathbb{R}^{d \times h} \quad b_z \in \mathbb{R}^d \tag{9}$$

The attention weights $\alpha_{n,i}^l$ are computed by a dot product of the encoder output vectors $e_1, \cdots, e_m$ with z_n^l and normalized by a softmax:

$$\alpha_{n,i}^l = \frac{\exp(e_i^T z_n^l)}{\sum_{k=1}^m \exp(e_k^T z_n^l)} \qquad i = 1, \cdots, m \tag{10}$$

The addition of the source embeddings helps to better retain information about the source tokens. The conditional source context vector x_n^l is a weighted sum of the encoder outputs as well as the source embeddings:

$$x_n^l = \sum_{i=1}^m \alpha_{n,i}^l (e_i + s_i) \tag{11}$$

The context vector x_n^l is then linearly mapped to $c_n^l \in \mathbb{R}^h$. The output vector of the l^{th} decoder layer, g_n^l, is the summation of c_n^l, y_n^l, and the previous layer's output vector g_n^{l-1}.

$$g_n^l = y_n^l + c_n^l + g_n^{l-1} \tag{12}$$

The final decoder layer output vector g_n^L is linearly mapped to $d_n \in \mathbb{R}^d$. Dropout [18] is applied at the decoder outputs, embeddings, and before every encoder and decoder layer. Finally, we compute a distribution over the T possible next target elements y_{i+1} by transforming the top decoder output d_n via a linear layer with weights W_o and bias b_o:

$$p(y_{i+1}|y_1, \cdots, y_i, S) = softmax(W_o d_n + b_o) \in \mathbb{R}^T \tag{13}$$

4 Experimental Setup

4.1 Data

We conduct our experiment on the dataset of NLPCC 2018 Evaluation Task 2, which is collected from Lang-8 website[2], a multilingual language learning platform providing language exchange Social Networking Service, with native speakers from more than 190 countries and 90 languages.

The full dataset, containing 1,220,069 sentence pairs, has no validation set. We randomly split the whole dataset into two parts: a validation set with 5k sentence pairs that have inconsistency between the source sentence and the target sentence and a training set with all the remaining 1,215,876 sentence pairs. In our experiments, we found that adding the sentence pairs that are identical on both sides could improve the result. Therefore we add all sentences with no grammatical error into the training set. The test data contains 2k sentence pairs. The statistic of the dataset shows in Table 1.

Table 1. Statistics of training, validation and test data. Unchanged refers to unchanged sentence pair. Changed refers to changed sentence pair. Src refers to source wrong sentences. Trg refers to target correct sentences.

	Unchanged	Changed	Words (Jieba)	Characters
Training Src	123,500	1,091,569	15,532,349	25,102,706
Training Trg	123,500	1,091,569	16,261,275	26,318,823
Validation Src	–	5,000	63,974	103,543
Validation Trg	–	5,000	67,342	108,965
Test Src	–	2,000	37,420	61,314

[2] http://lang-8.com/.

Table 2. Examples of two word segmentation methods.

The result of jieba tool	The result of subword method
这个/游戏/叫龙/和/地下城/。	这个/游戏/叫@@/龙/和/地下@@/城/。

4.2 Data Preparation

Word Segmentation. Since the evaluation criteria is based on the word-level, we firstly segment the large corpus using jieba[3] toolkit, which is a Python module for Chinese word segmentation. As is well known, Chinese word segmentation constantly faces the difficulty of multi-granularities. Remarkably, jieba deals with this kind of problem with flying colors. By comparing the experimental results of word segmentation with other word segmentation tools, we found that using jieba can achieve superior performance.

Subword. The subword method is initially proposed by Byte Pair Encoding (BPE) which is an effective data compression technique. Sennrich et al. [17] adopted BPE for word segmentation in neural machine translation (NMT) task which helps to solve the problem of rare and unknown words. The task of grammatical error correction has the similar problem like out-of-vocabulary (OOV) words in the summary generation. Hence, we apply this BPE algorithm to our task, which splits rare words into multiple frequent subwords. The results of the two segmentation methods are showed in Table 2. In our experiments, the use of BPE algorithm can greatly enhance the performance of the model. For detailed comparison results, see Sect. 4.3.

Word Embeddings. Word representations learned from large corpus have shown to be beneficial in many NLP tasks, such as part-of-speech tagging, dependency parsing and machine translation. We initialize the word embeddings for the source and target words with pre-trained word embeddings learned from a unlabeled large corpors. Word segmented by jieba tool, and rare words in this Chinese corpus are split into subword units by Byte Pair Encoding algorithm as we use similar preprocessing for the training dataset that is used to train the network. We use the structured-skipngram model in Wang2Vec tool [11] to train word vectors, which can solve syntax problems well and have information about the words order. In our experiments, the use of pre-trained word embedding can greatly enhance the performance of the model than initializing the network randomly. For detailed comparison results, see Sect. 4.3.

4.3 Experiment Results

We adopt the widely used MaxMatch Scorer toolkit for evaluation. Table 3 shows the results. Our basic model (CS2S) with no use of any additional knowledge

[3] https://github.com/fxsjy/jieba.

or strategy achieves 18.11 in $F_{0.5}$. The $F_{0.5}$ score increase to 20.11 with the utilization of pre-trained word embedding. By adapting the BPE algorithm to the preparation of the dataset, the performance boosts by 9.69 in $F_{0.5}$ (from 18.11 to 27.80). The results are consistent with our intuition that the BPE is supportive to the seq2seq model by upgrading its ability to generate unknown words.

Led by the previous experiments, we equip our model with both pre-trained embedding and BPE algorithm (CS2S+BPE+Emb). This model achieves 29.02 in $F_{0.5}$. Last but not least, we initialize the parameters of the fully equipped model with 4 different seeds. By ensembling the 4 models saved with different initializations, our approach achieves an $F_{0.5}$ score of 30.57, surpassing the best published result of 29.91 in $F_{0.5}$ (TeamID is Fighter_Plane) previously. Due to time suppress, we couldn't submit the results of the ensembled model.

Compared to English GEC, the best $F_{0.5}$ score we gain is an unsatisfying 30.57. To some extent, it lies on the scale of task and the deficiency of training data. So there is much to be explored for the task of Chinese GEC.

Table 3. Results on test dataset. +BPE refers to using byte pair encoding algorithm to preprocess data. +Emb indicates of using pre-trained embedding. Ensemble refers to merging results of 4 models with different initialization.

System	P	R	$F_{0.5}$
CS2S	21.28	11.36	18.11
CS2S+Emb	23.22	13.10	20.11
CS2S+BPE	40.27	12.90	27.80
CS2S+BPE+Emb	41.73	13.08	**29.02**
CS2S+BPE+Emb (ensemble)	47.63	12.56	**30.57**

5 Related Works

Grammatical Error Detection and Correction in CONLL-2013 and CONLL-2014 shared Task attracted a lot of English NLP researchers. Many different approaches were proposed by those participants, e.g. hand-crafted rules, statistical model, translation model and language model.

Statistical machine translation [8] has achieved good results, which can correct various types of errors and complex error patterns. However, SMT-based systems suffer from limited generalization capabilities compared to neural approaches and are unable to access longer source and target contexts effectively. To address these issues, several seq2seq approaches relying on RNNs were proposed for GEC.

At present, encoder-decoder frameworks are widely used for tasks like machine translation. Yuan et al. [19] first applied a popular neural machine translation model, *RNNSearch*. Ji et al. [7] proposed a hybrid word-character model based on the hybrid machine translation model, by adding nested levels of attention at the word and character level. More recently, Schmaltz et al. [16] used a word-level bidirectional LSTM network trained on Lang-8 and NUCLE with edit operations (insertions, deletions, and substitutions) marked with special tags in the target sentences.

More recently, Gehring et al. [6] propose an architecture for sequence to sequence modeling that is entirely convolutional. The model is equipped with gated linear units and residual connections, and also use attention in every decoder layer and demonstrate that each attention layer only adds a negligible amount of overhead. It performs well on some published dataset, because convolutional networks do not depend on the computations of the previous time step and therefore allow parallelization over every element in a sequence, so training and decoding speed is faster.

Due to the similarity between MT and GEC illustrated above, an encoder-decoder model can also be employed for the latter, where the encoder network is used to encode the potentially erroneous source sentence in vector space and a decoder network generates the corrected output sentence by attending to the output of the encoder stack.

6 Conclusion and Future Work

This paper describes our system in the NLPCC 2018 Shared Task 2 for GEC. We explored a seq2seq model based entirely on convolutional neural network. The application of BPE-based algorithm to split rare words into multiple frequent subwords makes the GEC model more capable of handling OOV problem. We achieved highest precision scores and $F_{0.5}$ score is 30.57.

At this stage, we believe the task is far from solved. Lots of improvements can be made to our current model. In the future, we will continue to work on this problem. Possible future directions include combining grammatical error correction with other related multi-task models, adding more features to the model and adapting pre-trained language model. Aside from the model architecture, due to the flexibility and intricacy of Chinese grammar, how to evaluate the automatic grammatical correction also remains a big challenge. In our future work, we will investigate better measurements and criteria for evaluation. Our code is released at https://github.com/blcu-nlp/NLPCC_2018_TASK2_GEC.

Acknowledgment. This research project is supported by the funds of Beijing Advanced Innovation Center for Language Resources (No. TYR17001), the Key Project of National Social Science Foundation of China (No. 16AYY007), and the Fundamental Research Funds for the Central Universities in BLCU (No. 18YCX001). We also thank Huimeng Zhang, JingJing Miao and Jin Zhao for helping to modify the paper.

References

1. Chollampatt, S., Ng, H.T.: Connecting the dots: towards human-level grammatical error correction. In: BEA@EMNLP (2017)
2. Chollampatt, S., Ng, H.T.: A multilayer convolutional encoder-decoder neural network for grammatical error correction. CoRR abs/1801.08831 (2018)
3. Chorowski, J., Bahdanau, D., Cho, K., Bengio, Y.: End-to-end continuous speech recognition using attention-based recurrent NN: first results. CoRR abs/1412.1602 (2014)
4. Dahlmeier, D., Ng, H.T.: Better evaluation for grammatical error correction. In: NAACL (2012)
5. Dauphin, Y., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: ICML (2017)
6. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.: Convolutional sequence to sequence learning. In: ICML (2017)
7. Ji, J., Wang, Q., Toutanova, K., Gong, Y., Truong, S., Gao, J.: A nested attention neural hybrid model for grammatical error correction. In: ACL (2017)
8. Junczys-Dowmunt, M., Grundkiewicz, R.: Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In: EMNLP (2016)
9. Junczys-Dowmunt, M., Grundkiewicz, R., Guha, S., Heafield, K.: Approaching neural grammatical error correction as a low-resource machine translation task. CoRR abs/1804.05940 (2018)
10. Yu, L.-C., Lee, L.H., Chang, L.P.: Overview of grammatical error diagnosis for learning Chinese as foreign language. In: NLP-TEA (2014)
11. Ling, W., Dyer, C., Black, A.W., Trancoso, I.: Two/too simple adaptations of word2vec for syntax problems. In: NAACL (2015)
12. Neubig, G.: Neural machine translation and sequence-to-sequence models: a tutorial. CoRR abs/1703.01619 (2017)
13. Ng, H.T., Wu, S.M., Wu, Y., Hadiwinoto, C., Tetreault, J.R.: The CoNLL-2013 shared task on grammatical error correction. In: CoNLL Shared Task (2013)
14. Rao, G., Zhang, B., Xun, E., Lee, L.H.: IJCNLP-2017 task 1: Chinese grammatical error diagnosis. In: IJCNLP (2017)
15. Rush, A.M., Chopra, S., Weston, J.: A neural attention model for abstractive sentence summarization. In: EMNLP (2015)
16. Schmaltz, A., Kim, Y., Rush, A.M., Shieber, S.M.: Adapting sequence models for sentence correction. In: EMNLP (2017)
17. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. CoRR abs/1508.07909 (2016)
18. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR **15**, 1929–1958 (2014)
19. Yuan, Z., Briscoe, T.: Grammatical error correction using neural machine translation. In: NAACL (2016)