Chinese Grammatical Error Correction Based on Hybrid Models with Data Augmentation

Yi Wang\textsuperscript{1,3,*}, Ruibin Yuan\textsuperscript{2,*}, YanGen Luo\textsuperscript{1,3,*}, YuFang Qin\textsuperscript{1,3,*}, NianYong Zhu\textsuperscript{1,3}, Peng Cheng\textsuperscript{1,3}, and Lihuan Wang\textsuperscript{1,3}

\textsuperscript{1}State Key Laboratory of Media Convergence Production Technology and Systems Xinhua News Agency, Beijing, 100077, China.
\textsuperscript{2}Stardust.ai, Beijing, China.
\textsuperscript{3}The Technical Bureau of Xinhua News Agency

*These authors contributed to the work equally and should be regarded as co-first authors.

Abstract

A better Chinese Grammatical Error Diagnosis (CGED) system for automatic Grammatical Error Correction (GEC) can benefit foreign Chinese learners and lower Chinese learning barriers. In this paper, we introduce our solution to the CGED2020 Shared Task Grammatical Error Correction in detail. The task aims to detect and correct grammatical errors that occur in essays written by foreign Chinese learners. Our solution combined data augmentation methods, spelling check methods, and generative grammatical correction methods, and achieved the best recall score in the Top 1 Correction track. Our final result ranked fourth among the participants.

1 Introduction

In recent years, a global upsurge of Chinese learning has been set off. However, due to the language environment and language structure differences between countries, foreign Chinese learners are more prone to grammatical errors. Traditional grammatical error correction mainly relies on rule-based methods and performs poorly. Therefore, a better Chinese grammar diagnosis system is needed. Thanks to NLPTEA, the Chinese Grammatical Error Diagnosis (CGED) shared task provides a free communication platform for computing technology researchers in natural language processing (NLP) to seek more advanced Chinese grammar diagnosis solutions.

Due to the deficiency of parallel corpora, Chinese GEC often used statistical methods and rule-based methods in the early stage. Until recently, with larger-scale parallel corpora developed, machine learning techniques were applied to the Chinese GEC task. Chen (Zheng et al., 2016) used an approach based on the conditional random fields (CRF) model. The model added a collocation feature in order to better identify grammatical errors in word choice. Zheng (Zheng et al., 2016) used a CRF based model, along with an RNN based model and an ensemble model, reached high F1-scores and recall rates across the three assessment levels of the NLP-TEA-3 shared task. Yang (Yang et al., 2017) leveraged a bi-LSTM-CRF model. Spliced word vectors with manual features such as bi-gram, POS, and PMI were added during training. Fu (Fu et al., 2018) also used a bi-LSTM-CRF model, with ePMI values integrated. They obtained promising results in the CGED2018 shared
task.

In this paper, we introduce our solution to the CGED2020 shared tasks. By combining data augmentation methods, spelling check methods, and generative grammatical correction methods, we achieved the best recall score in the Top 1 Correction track. Our final result ranked fourth among the participants. The rest of this article is organized as follows: Section 2 describes the shared tasks of CGED2020. Section 3 describes the methods used in this paper, including data preprocessing, data augmentation, and various deep learning error correction models. Section 4 conducts experiments on the methods mentioned above. In Section 5, the conclusion and the planning for future works are given.

2 Task Definition

The CGED2020 shared task is the sixth Chinese grammar diagnosis error competition, held since 2014. This task provides participants a shared data set using the writing part of Hanyu Shuiping Kaoshi (HSK). The goal and direction of the task are to use modern NLP techniques to detect foreign Chinese learners’ grammatical errors in Chinese writing and build an automatic Chinese grammatical error diagnosis system. It mainly distinguishes four different types of errors, including Redundant Words (R), Missing Words (M), Word Selection (S), and Word Order Error (W). On this basis, a comprehensive evaluation is carried out according to these dimensions of error judgment of the problem sentence, error type analysis, the error location, and sentence modification suggestions. The detailed sample is shown in Table 1.

The criteria for judging correctness are determined at three levels as follows.

(1) Detection-level: Binary classification of a given sentence, correct or incorrect, should be completely identical with the gold standard. All error types will be regarded as incorrect.

(2) Identification-level: This level could be considered as a multi-class categorization problem. All error types should be identified according to the gold standard.

(3) Position-level: In addition to identifying the error types, this level also judges the grammatical error’s occurrence range.

3 Methodology

3.1 Data Preparation

We use the dataset from Ren (Ren et al., 2018), which contains 1.3 million sentence pairs collected from Lang-8 and HSK. Native Chinese speakers wrote news articles published by the Xinhua News Agency during 2017 and 2018, and compositions are collected for data augmentation. The former contains 6 million sentence pairs, and the latter contains 1 million sentence pairs. Texts are split into sentences, and all the non-Chinese, non-English, and non-punctuation characters in the sentences are removed. Also, sentences that are longer than 64 characters are discarded. Finally, we randomly choose 10000 pairs of sentences from non-augmented data and equally split them into validation and testing sets. The rest is used for training.

3.2 Data Augmentation

Obtaining adequate parallel data for deep learning-based GEC models is a challenging task, especially in the Chinese language. To mitigate the problem, a data augmentation scheme is applied. In this paper, we combine both rule-based and neural network-based
methods for generating noisy, ungrammatical texts from their clean counterparts. After the augmentation process, 6 million pairs of rule-based and 1 million pairs of neural network-based clean-corrupted sentences are obtained for training.

### 3.2.1 Rule Based Corruption

Inspired by previous work by Wang (Wang et al., 2019), we propose a rule-based corpora corruption method. Unlike Wang’s method, our method performs both word grain and character grain corruption and introduces sentence grain word ordering error to corpora. This method aims to obtain a large amount of parallel data with rich, diverse errors within a short time.

According to Wang, imbalanced error types will lead to low recall on the low-frequency error types. Therefore, the probability of each artificial error type are set to equal. As using low error rate data for training will cause the model to become too conservative, the corruption rate $P_{corrupt}$ is set to 0.4. With a sentence given, we obtain both character grain tokens $t_c$ and word grain tokens $t_w$ (using jieba). At each step, we corrupt a $t_c$ or a $t_w$ with a probability of $P_{corrupt}$. The corruption operations include, inserting a random character $c_{rand}$ in the vocabulary $V$ or a synonym $syn$ (using synonyms) to the left of a token with a probability of $p_r$ (redundant error type), replacing a token with a random character $c_{rand}$ in $V$ or a low similarity synonym $syn_{low}$ with a probability of $p_s$ (selection error type), deleting a token with a probability of $p_m$ (missing error type), moving a token to a random position with a probability of $p_w$ (word ordering error type). Algorithm 1 formalizes this method and Table 2 shows the corrupted results.

### 3.2.2 Neural Network Based Corruption

We train an attention-based sequence-to-sequence model with a bidirectional GRU encoder to generate noisy counterparts for clean sentences. This approach aims to generate realistic ungrammatical parallel corpora from clean corpora but is limited by the inference speed. Borrowing ideas from but being different from Xie (Xie et al., 2018) which used a noisy beam search scheme to introduce noise into the decoding stage, we define noisy score $s_{noisy}$ as:

$$s_{noisy} = s - \tau \beta_{random}$$

where $s$ is the log-probability of a token, $\beta_{random}$ is a scale factor and $\tau$ is a uniform random variable $\tau \sim U(0, 1)$. We use the beam size of 6 to balance between the decod-
Algorithm 1: Rule Based Corpora Corruption Method

Input: vocabulary \( V \), clean sentences corpora \( C_{\text{clean}} \), corruption rate \( P_{\text{corrupt}} \), probability of redundant error type \( p_r \), probability of selection error type \( p_s \), probability of missing error type \( p_m \), probability of word ordering error type \( p_w \), synonym \( \text{syn}() \) generator

Output: corrupted corpora \( C_{\text{noisy}} \)

Initialize \( C_{\text{noisy}} = {} \)

for each sentences \( s \) in \( C_{\text{clean}} \) do

\( \text{rand} = \text{Random}(0,1) \)

if \( \text{rand} < P_{\text{corrupt}} \times p_w \) then

move a random word grain token \( t_w \) to a random position

end

\( \text{rand} = \text{Random}(0,1) \)

if \( \text{rand} > P_{\text{corrupt}} \) then

continue

end

\( \text{rand} = \text{Random}(0,1) \)

if \( \text{rand} < 0.5 \) then

for each character grain token \( t_c \) in \( s \) do

\( \text{rand} = \text{Random}(0,1) \)

if \( \text{rand} < p_r \) then

insert a token \( \text{rand} \) in \( V \) to the left of \( t_c \)

else if \( \text{rand} < p_r + p_s \) then

replace \( t_c \) with a token \( \text{rand} \) in \( V \)

else if \( \text{rand} < p_r + p_s + p_m \) then

delete \( t_c \) from \( s \)

end if

end

else

for each word grain token \( t_w \) in \( s \) do

\( \text{rand} = \text{Random}(0,1) \)

if \( \text{rand} < p_r \) then

insert a synonym \( \text{syn} = \text{syn}(t_w) \) to the left of \( t_w \)

else if \( \text{rand} < p_r + p_s \) then

replace \( t_w \) with a low similarity synonym \( \text{syn}_{\text{low}} = \text{syn}(t_w) \)

else if \( \text{rand} < p_r + p_s + p_m \) then

delete \( t_w \) from \( s \)

end if

end

end if

add \( s \) to \( C_{\text{noisy}} \)
end

return \( C_{\text{noisy}} \)
不过，特朗普无视礼仪，语出惊人，频戳痛点，不仅双边关系没能拉近，反而平添几分不和谐。

不过，特朗普无视礼仪，语出惊人，频戳痛点，不仅双边关系没能拉近，反而平添几分不和谐。

表2: 基于规则的错误结果。

Table 2: Rule based corruption results.

3.3 生成模型

我们采用两种生成模型，Lasertagger和Conv-Seq2Seq，用于语法错误纠正。

Lasertagger由Malmi (Malmi et al., 2019)提出，采用序列标记方法进行GEC。它将GEC问题转化为文本编辑任务，因为错误序列和其修正的编辑距离总是很低。该方法主要结合了BERT编码器和自回归Transformer解码器来预测三个主要编辑操作：保持一个词，删除一个词，以及在词之前添加一个短语。我们使用Lasertagger AR与RoBERTa (Cui et al., 2019)作为骨干，将词表大小设置为1000，在短语提取阶段。其他参数设置为默认。

Conv-Seq2Seq由Gehring (Gehring et al., 2017)提出，是一个CNN基于的序列到序列 (Seq2Seq) 模型。它将GEC问题看作是机器翻译任务，并将有噪的、不规范的句子翻译成干净的句子。与基于RNN的Seq2Seq模型不同，它在训练阶段更有效率，并且对局部n-gram特征更敏感。Ren等 (Ren et al., 2018)的工作已经证明了它的有效性。我们在与Ren同样的设置下进行训练。

3.4 拼写检查模型

我们还采用拼写检查模型，SpellGCN和Electra，用于更好地处理拼写错误。注意，当应用拼写检查器到GEC时，我们使用一种后处理策略来过滤假阳性样本，包括对人名、数字或“他他它”等的修改，以及使目标句子段落长度大于原始句子段落长度的修改。

SpellGCN (Cheng et al., 2020)将每个相似字符集分为两个相似性图，一个用于相似发音，另一个用于相似拼写。
Table 3: Neural network-based corruption results.

| Input: <go> 可能会感到快乐，<eos>
| Predicted: 歌曲使人感到快乐，<eos>

| Input: <go> 现在知道你讲中文很风趣了！<eos>
| Predicted: 今天我知道你的讲中文很风趣！<eos>

| Input: <go> 你是哪里人？<eos>
| Predicted: 你是哪里人？<eos>

| Input: <go> 手下们累得气喘吁吁的时候，那位老人的身份被公开了。<eos>
| Predicted: 手下都累得气喘吁吁的时候，那位老人被身分公开了。<eos>

| Input: <go> 我们一起玩了一些时候，然后吃了晚饭。<eos>
| Predicted: 所以我们一起玩了一些时间，然后就吃晚饭。<eos>

| Input: <go> 关于中国政府奖学金申请，我有几个问题希望得到您的解答。<eos>
| Predicted: 有关于中国政府奖学金申请，我有几个问题希望得到您的答应。<eos>

| Input: <go> 我害怕如果我们没有进展，那样我会很难过 <eos>
| Predicted: 我害怕如果我们不可以进展，不过我很难过 <eos>

| Input: <go> 考虑到韩国的老年贫困率和老年自杀率位居世界第一这一点，情况就特别严重。<eos>
| Predicted: 考虑韩国的老年贫困率老年的杀害率位于世界第一这一点，情况特别严重。<eos>

| Input: <go> 尽管我有了非常多空闲，但我还没写完我答应要发送给你的故事。<eos>
| Predicted: 无论我有非常多空，我还没写完我答应要送给你你的故事。<eos>

for similar shape. Then it takes the graphs as input and generates an embedding for each character after the interaction between similar characters. These embeddings are then constructed into a character classifier for the semantic representation extracted from another backbone module. With the Combination of graph representation and BERT, SpellGCN can leverage the similarity knowledge and generate the right corrections accordingly. We use the default setting as Cheng, and a fine tuned BERT by Xu(Ming, 2020) as the backbone.

Electra(Clark et al., 2020) is a pre-training language model with a new pre-training task and framework, which changed the generative masked language model (MLM) pre-training task into the discriminant replaced token detection (RTD) task to determine whether the current token has been replaced by the language model. Experiments of paper show that the context representation learned by Electra is much better than the context representation learned by Bert and XLnet under the same model size, data, and calculation conditions. We use the Chinese version Electra-base model released by iFLYTEK Joint Laboratory of Harbin Institute of technology for spelling check.

3.5 N-Best Reranker

With the mentioned spelling check models and generative models, we use a recursive method for Chinese GEC as shown in Figure 1. For a given input sentence, it will first go through two spelling check models in a parallel fashion. Then the post spelling check output with the original sentence will go through the generative models, also in parallel. The whole process can loop for K (K=3) rounds to obtain N (N=6) output sentences. A MERT reranker is deployed to rerank N sentences, and we select the top-1 sentence as the correction result. Note that we use a post processing script to transform the result to present the detection and position level result. Several features are introduced during reranking: 1. Normalized 4-gram language model score divided by sentence length, 2. Edit operations including character add/delete/swap count from
source sentence to target sentence, 3. mask-predicted Bert probability score of a target sentence (Chollampatt et al., 2019), 4. Target sentence length penalty. The reranker is based on MERT from Moses (Koehn et al., 2007), with tuning metric of M2 F1-score.

Also, a more simple reranking approach is deployed to choose the best sentence according to the mask-predicted Bert score, which is computed by:

$$S_{BERT}(Y) = \sum_{i=1}^{[Y]} \log P_{BERT}(y_{i}|Y_{i-maked})$$

where $Y$ is the target sentence, $Y_{i-maked}$ is the target sentence with $i$-th word $y_i$ being masked, $P_{BERT}$ is the Bert output probability.

4 Experiments

We first train Lasertagger on HSK+Lang8 dataset to obtain LASER_RAW. Then we add spelling checkers to obtain LASER_RAW+SPELL, which boosts each score by about 0.03. We also train a Lasertagger with augmented data only to obtain LASER_AUG. The promising result shows the effectiveness of our data augmentation methods. After adding spelling checkers, denoted as LASER_AUG+SPELL, we obtain the best correction score, which is 0.1993. Also, we try to pretrain a Lasertagger on augmented data for 10 epoch and fine tune on real data (HSK+Lang8), denoted by LASER_FINE. The model performs better than LASER_RAW and LASER_AUG on detection level and identification level, but worse in correction level. With spelling checkers adding in, we obtain LASER_FINE+SPELL, with a similar boosting effect as previous results. Finally, we add in Conv-Seq2Seq and the two rerankers and denoted as MERT_RERANK and SIMPLE_RERANK. We observe a boosting effect on each level except for a significant downgrade in correction level when using MERT. We submitted the MERT_RERANK as the final result. The testing results are shown in Table 4.

We first train Lasertagger on HSK+Lang8 dataset to obtain LASER_RAW. Then we add spelling checkers to obtain LASER_RAW+SPELL, which boosts each score by about 0.03. We also train a Lasertagger with augmented data only to obtain LASER_AUG. The promising result shows the effectiveness of our data augmentation methods. After adding spelling checkers, denoted as LASER_AUG+SPELL, we obtain the best correction score, which is 0.1993. Also, we try to pretrain a Lasertagger on augmented data for 10 epoch and fine tune on real data (HSK+Lang8), denoted by LASER_FINE. The model performs better than LASER_RAW and LASER_AUG on de-
Table 4: Testing results on CGED2020 testing set.

| EXPS                      | Detection | Identification | Position | Correction Top1 |
|---------------------------|-----------|----------------|----------|-----------------|
| LASER_RAW                 | 0.8258    | 0.5080         | 0.2375   | 0.1332          |
| LASER_RAW+SPELL           | 0.8517    | 0.5410         | 0.2649   | 0.1668          |
| LASER_AUG                 | 0.8221    | 0.5319         | 0.2595   | 0.1826          |
| LASER_AUG+SPELL           | 0.8475    | 0.5672         | 0.2799   | 0.1993          |
| LASER_FINE                | 0.8597    | 0.5610         | 0.2531   | 0.1602          |
| LASER_FINE+SPELL          | 0.8731    | 0.5837         | 0.2727   | 0.1857          |
| MERT_RERANK               | 0.8852    | 0.6203         | 0.2812   | 0.1683          |
| SIMPLE_RERANK             | 0.8846    | 0.5966         | 0.3009   | 0.1976          |

tection level and identification level, but worse in correction level. With spelling checkers adding in, we obtain LASER_FINE+SPELL, with similar boosting effect as previous results. Finally, we add in Conv-Seq2Seq and the two rerankers and denoted as MERT_RERANK and SIMPLE_RERANK. We observe a boosting effect in each level except for a significant down grade in correction level when using MERT. We submitted the MERT_RERANK as final result. Testing results are shown in Table 2.

Since our pipeline does not work well in the FPR track (FPR ≥ 0.7068) and performance downgrade is observed, we look into the MERT reranker results. We find that the 4-gram model does not perform well. It tends to give shorter sentences higher scores, and it was trained on news domain data, so domain adaption can be an issue.

5 Conclusion and Future Works

This paper describes our system in the CGED2020 Shared Task Grammatical Error Correction. We explored a scheme by combining data augmentation methods, spelling check methods, and generative grammatical correction methods. We achieved the best recall score and our final result ranked fourth. However, there are still many efforts needed to solve this problem. A lot of improvements can be made to our current model. In the future, we will continue working on this problem. Possible future directions include improving data augmentation methods, finding a better reranking strategy, and finding better measurements for evaluation.

References

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. Spellgcn: Incorporating phonological and visual similarities into language models for chinese spelling check.

Shamil Chollampatt, Weiqi Wang, and Hwee Ng. 2019. Cross-sentence grammatical error correction. pages 435–445.

Kevin Clark, Minh Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert.

Ruiji Fu, Zhengqi Pei, Jiefu Gong, Wei Song, Dechuan Teng, W. Che, S. Wang, G. Hu, and T. Liu. 2018. Chinese grammatical error diagnosis using statistical and prior knowledge driven features with probabilistic ensemble enhancement. In NLP-TEA@ACL.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer,
Ondřej Bojar, Alex Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing.

Xu Ming. 2020. pycorrector: Text correction tool [software].

Hongkai Ren, Liner Yang, and Endong Xun. 2018. A sequence to sequence learning for chinese grammatical error correction. In CCF International Conference on Natural Language Processing and Chinese Computing.

Chencheng Wang, Liner Yang, Yun Chen, Yongping Du, and Erhong Yang. 2019. Controllable data synthesis method for grammatical error correction.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse back-translation for grammar correction. pages 619–628.

Yi Yang, Pengjun Xie, J. Tao, Guangwei Xu, L. Li, and S. Luo. 2017. Alibaba at ijcnlp-2017 task 1: Embedding grammatical features into lstms for chinese grammatical error diagnosis task. In IJCNLP.

Bo Zheng, W. Che, Jiang Guo, and T. Liu. 2016. Chinese grammatical error diagnosis with long short-term memory networks. In NLP-TEA@COLING.