

Tail-to-Tail Non-Autoregressive Sequence Prediction for Chinese Grammatical Error Correction

Piji Li Shuming Shi

Tencent AI Lab, Shenzhen, China

{pijili, shumingshi}@tencent.com

Abstract

We investigate the problem of Chinese Grammatical Error Correction (CGEC) and present a new framework named Tail-to-Tail (TtT) non-autoregressive sequence prediction to address the deep issues hidden in CGEC. Considering that most tokens are correct and can be conveyed directly from source to target, and the error positions can be estimated and corrected based on the bidirectional context information, thus we employ a BERT-initialized Transformer Encoder as the backbone model to conduct information modeling and conveying. Considering that only relying on the same position substitution cannot handle the variable-length correction cases, various operations such substitution, deletion, insertion, and local paraphrasing are required jointly. Therefore, a Conditional Random Fields (CRF) layer is stacked on the up tail to conduct non-autoregressive sequence prediction by modeling the token dependencies. Since most tokens are correct and easily to be predicted/conveyed to the target, then the models may suffer from a severe class imbalance issue. To alleviate this problem, focal loss penalty strategies are integrated into the loss functions. Moreover, besides the typical fix-length error correction datasets, we also construct a variable-length corpus to conduct experiments. Experimental results on standard datasets, especially on the variable-length datasets, demonstrate the effectiveness of TtT in terms of sentence-level Accuracy, Precision, Recall, and F1-Measure on tasks of error Detection and Correction¹.

1 Introduction

Grammatical Error Correction (GEC) aims to automatically detect and correct the grammatical errors that can be found in a sentence (Wang et al., 2020c). It is a crucial and essential application task

Correct	今天我感到非常高兴! I feel very happy today!
Type I	今天我感到飞长高兴! I feel fly long happy today! <small>非常</small>
Type II	今天我到非常常高兴! I am always happy when I come to Fei today! <small>感</small>
Type III	今天我非常感到高兴! I very feel happy today!

Figure 1: Illustration for the three types of operations to correct the grammatical errors: Type I-substitution; Type II-deletion and insertion; Type III-local paraphrasing.

in many natural language processing scenarios such as writing assistant (Ghufron and Rosyida, 2018; Napoles et al., 2017; Omelianchuk et al., 2020), search engine (Martins and Silva, 2004; Gao et al., 2010; Duan and Hsu, 2011), speech recognition systems (Karat et al., 1999; Wang et al., 2020a; Kubis et al., 2020), etc. Grammatical errors may appear in all languages (Dale et al., 2012; Xing et al., 2013; Ng et al., 2014; Rozovskaya et al., 2015; Bryant et al., 2019), in this paper, we only focus to tackle the problem of *Chinese Grammatical Error Correction (CGEC)* (Chang, 1995).

We investigate the problem of CGEC and the related corpora from SIGHAN (Tseng et al., 2015) and NLPC (Zhao et al., 2018) carefully, and we conclude that the grammatical error types as well as the corresponding correction operations can be categorised into three folds, as shown in Figure 1: (1) **Substitution**. In reality, Pinyin is the most popular input method used for Chinese writings. Thus, the homophonous character confusion (For example, in the case of Type I, the pronunciation of the wrong and correct words are both “FeiChang”) is the fundamental reason which causes grammatical errors (or spelling errors) and can be corrected by substitution operations without changing the whole sequence structure (e.g., length). Thus, substitution is a *fixed-length (FixLen) operation*. (2) **Deletion**

¹Code: <https://github.com/lipiji/TtT>



Figure 2: Illustration of the token information flows from the bottom tail to the up tail.

and **Insertion**. These two operations are used to handle the cases of word redundancies and omissions respectively. (3) **Local paraphrasing**. Sometimes, light operations such as substitution, deletion, and insertion cannot correct the errors directly, therefore, a slightly subsequence paraphrasing is required to reorder partial words of the sentence, the case is shown in Type III of Figure 1. Deletion, insertion, and local paraphrasing can be regarded as *variable-length (VarLen) operations* because they may change the sentence length.

However, over the past few years, although a number of methods have been developed to deal with the problem of CGEC, some crucial and essential aspects are still uncovered. Generally, sequence translation and sequence tagging are the two most typical technical paradigms to tackle the problem of CGEC. Benefiting from the development of neural machine translation (Bahdanau et al., 2015; Vaswani et al., 2017), attention-based seq2seq encoder-decoder frameworks have been introduced to address the CGEC problem in a sequence translation manner (Wang et al., 2018; Ge et al., 2018; Wang et al., 2019, 2020b; Kaneko et al., 2020). Seq2seq based translation models are easily to be trained and can handle all types of correcting operations above mentioned. However, considering the exposure bias issue (Ranzato et al., 2016; Zhang et al., 2019), the generated results usually suffer from the phenomenon of hallucination (Nie et al., 2019; Maynez et al., 2020) and cannot be faithful to the source text, even though copy mechanisms (Gu et al., 2016) are incorporated (Wang et al., 2019). Therefore, Omelianchuk et al. (2020) and Liang et al. (2020) propose to purely employ tagging to conduct the problem of GEC instead of generation. All correcting operations such as deletion, insertion, and substitution can be guided by the predicted tags. Nevertheless, the pure tagging strategy requires to extend

the vocabulary \mathcal{V} to about three times by adding “insertion-” and “substitution-” prefixes to the original tokens (e.g., “insertion-good”, “substitution-paper”) which decrease the computing efficiency dramatically. Moreover, the pure tagging framework needs to conduct multi-pass prediction until no more operations are predicted, which is inefficient and less elegant. Recently, many researchers fine-tune the pre-trained language models such as BERT on the task of CGEC and obtain reasonable results (Zhao et al., 2019; Hong et al., 2019; Zhang et al., 2020b). However, limited by the BERT framework, most of them can only address the fixed-length correcting scenarios and cannot conduct deletion, insertion, and local paraphrasing operations flexibly.

Moreover, during the investigations, we also observe an obvious but crucial phenomenon for CGEC that most words in a sentence are correct and need not to be changed. This phenomenon is depicted in Figure 2, where the operation flow is from the bottom tail to the up tail. Grey dash lines represent the “Keep” operations and the red solid lines indicate those three types of correcting operations mentioned above. On one side, intuitively, the target CGEC model should have the ability of directly moving the correct tokens from bottom tail to up tail, then Transformer (Vaswani et al., 2017) based encoder (say BERT) seems to be a preference. On the other side, considering that almost all typical CGEC models are built based on the paradigms of sequence tagging or sequence translation, Maximum Likelihood Estimation (MLE) (Myung, 2003) is usually used as the parameter learning approach, which in the scenario of CGEC, will suffer from a severe class/tag imbalance issue. However, no previous works investigate this problem thoroughly on the task of CGEC.

To conquer all above-mentioned challenges, we propose a new framework named tail-to-tail non-

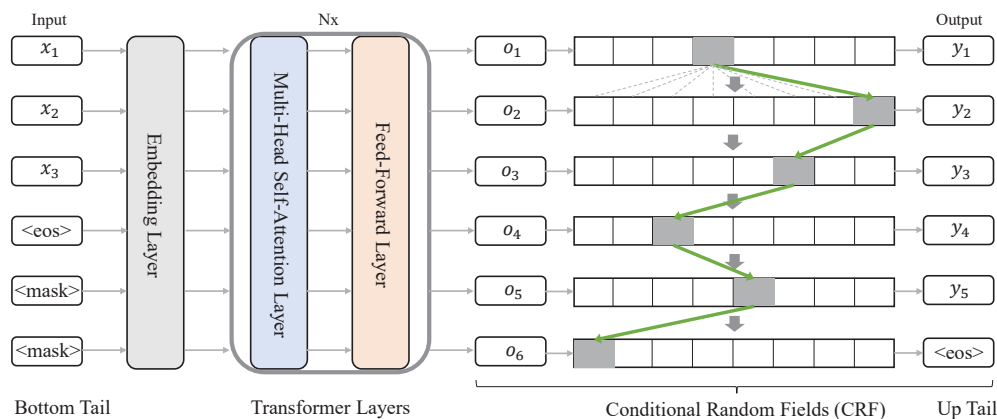


Figure 3: The proposed tail-to-tail non-autoregressive sequence prediction framework (TtT).

autoregressive sequence prediction, which abbreviated as **TtT**, for the problem of CGEC. Specifically, to directly move the token information from the bottom tail to the up tail, a BERT based sequence encoder is introduced to conduct bidirectional representation learning. In order to conduct substitution, deletion, insertion, and local paraphrasing simultaneously, inspired by (Sun et al., 2019), a Conditional Random Fields (CRF) (Lafferty et al., 2001) layer is stacked on the up tail to conduct non-autoregressive sequence prediction by modeling the dependencies among neighbour tokens. Focal loss penalty strategy (Lin et al., 2020) is adopted to alleviate the class imbalance problem considering that most of the tokens in a sentence are not changed. In summary, our contributions are as follows:

- A new framework named tail-to-tail non-autoregressive sequence prediction (TtT) is proposed to tackle the problem of CGEC.
- BERT encoder with a CRF layer is employed as the backbone, which can conduct substitution, deletion, insertion, and local paraphrasing simultaneously.
- Focal loss penalty strategy is adopted to alleviate the class imbalance problem considering that most of the tokens in a sentence are not changed.
- Extensive experiments on several benchmark datasets, especially on the variable-length grammatical correction datasets, demonstrate the effectiveness of the proposed approach.

2 The Proposed TtT Framework

2.1 Overview

Figure 3 depicts the basic components of our proposed framework TtT. Input is an incorrect sen-

tence $X = (x_1, x_2, \dots, x_T)$ which contains grammatical errors, where x_i denotes each token (Chinese character) in the sentence, and T is the length of X . The objective of the task grammatical error correction is to correct all errors in X and generate a new sentence $Y = (y_1, y_2, \dots, y_{T'})$. Here, it is important to emphasize that T is not necessary equal to T' . Therefore, T' can be $=$, $>$, or $<$ T . Bidirectional semantic modeling and bottom-to-up directly token information conveying are conducted by several Transformer (Vaswani et al., 2017) layers. A Conditional Random Fields (CRF) (Lafferty et al., 2001) layer is stacked on the up tail to conduct the non-autoregressive sequence generation by modeling the dependencies among neighboring tokens. Low-rank decomposition and beamed Viterbi algorithm are introduced to accelerate the computations. Focal loss penalty strategy (Lin et al., 2020) is adopted to alleviate the class imbalance problem during the training stage.

2.2 Variable-Length Input

Since the length T' of the target sentence Y is not necessary equal to the length T of the input sequence X . Then in the training and inference stage, different length will affect the completeness of the predicted sentence, especially when $T < T'$. To handle this issue, several simple tricks are designed to pre-process the samples. Assuming $X = (x_1, x_2, x_3, \langle \text{eos} \rangle)$: (1) When $T = T'$, i.e., $Y = (y_1, y_2, y_3, \langle \text{eos} \rangle)$, then do nothing; (2) When $T > T'$, say $Y = (y_1, y_2, \langle \text{eos} \rangle)$, which means that some tokens in X will be deleted during correcting. Then in the training stage, we can pad $T - T'$ special tokens $\langle \text{pad} \rangle$ to the tail of Y to make $T = T'$, then

$$Y = (y_1, y_2, \langle \text{eos} \rangle, \langle \text{pad} \rangle);$$

(3) When $T < T'$, say

$$Y = (y_1, y_2, y_3, y_4, y_5, \langle \text{eos} \rangle),$$

which means that more information should be inserted into the original sentence X . Then, we will pad the special symbol $\langle \text{mask} \rangle$ to the tail of X to indicate that these positions possibly can be translated into some new real tokens:

$$X = (x_1, x_2, x_3, \langle \text{eos} \rangle, \langle \text{mask} \rangle, \langle \text{mask} \rangle).$$

2.3 Bidirectional Semantic Modeling

Transformer layers (Vaswani et al., 2017) are particularly well suited to be employed to conduct the bidirectional semantic modeling and bottom-to-up information conveying. As shown in Figure 3, after preparing the input samples, an embedding layer and a stack of Transformer layers initialized with a pre-trained Chinese BERT (Devlin et al., 2019) are followed to conduct the semantic modeling.

Specifically, for the input, we first obtain the representations by summing the word embeddings with the positional embeddings:

$$\mathbf{H}_t^0 = \mathbf{E}_{w_t} + \mathbf{E}_{p_t} \quad (1)$$

where 0 is the layer index and t is the state index. \mathbf{E}_w and \mathbf{E}_p are the embedding vectors for tokens and positions, respectively.

Then the obtained embedding vectors \mathbf{H}^0 are fed into several Transformer layers. Multi-head self-attention is used to conduct bidirectional representation learning:

$$\begin{aligned} \mathbf{H}_t^1 &= \text{LN}(\text{FFN}(\mathbf{H}_t^1) + \mathbf{H}_t^1) \\ \mathbf{H}_t^1 &= \text{LN}(\text{SLF-ATT}(\mathbf{Q}_t^0, \mathbf{K}^0, \mathbf{V}^0) + \mathbf{H}_t^0) \\ \mathbf{Q}^0 &= \mathbf{H}^0 \mathbf{W}^Q \\ \mathbf{K}^0, \mathbf{V}^0 &= \mathbf{H}^0 \mathbf{W}^K, \mathbf{H}^0 \mathbf{W}^V \end{aligned} \quad (2)$$

where SLF-ATT(\cdot), LN(\cdot), and FFN(\cdot) represent self-attention mechanism, layer normalization, and feed-forward network respectively (Vaswani et al., 2017). Note that our model is a non-autoregressive sequence prediction framework, thus we use all the sequence states \mathbf{K}^0 and \mathbf{V}^0 as the attention context. Then each node will absorb the context information bidirectionally. After L Transformer layers, we obtain the final output representation vectors $\mathbf{H}^L \in \mathbb{R}^{\max(T, T') \times d}$.

2.4 Non-Autoregressive Sequence Prediction

Direct Prediction The objective of our model is to translate the input sentence X which contains grammatical errors into a correct sentence Y . Then, since we have obtained the sequence representation vectors \mathbf{H}^L , we can directly add a softmax layer to predict the results, just similar to the methods used in non-autoregressive neural machine translation (Gu and Kong, 2020) and BERT-based fine-tuning framework for the task of grammatical error correction (Zhao et al., 2019; Hong et al., 2019; Zhang et al., 2020b).

Specifically, a linear transformation layer is plugged in and softmax operation is utilized to generate a probability distribution $P_{\text{dp}}(y_t)$ over the target vocabulary \mathcal{V} :

$$\begin{aligned} \mathbf{s}_t &= \mathbf{h}_t^\top \mathbf{W}_s + \mathbf{b}_s \\ P_{\text{dp}}(y_t) &= \text{softmax}(\mathbf{s}_t) \end{aligned} \quad (3)$$

where $\mathbf{h}_t \in \mathbb{R}^d$, $\mathbf{W}_s \in \mathbb{R}^{d \times |\mathcal{V}|}$, $\mathbf{b}_s \in \mathbb{R}^{|\mathcal{V}|}$, and $\mathbf{s}_t \in \mathbb{R}^{|\mathcal{V}|}$. Then we obtain the result for each state based on the predicted distribution:

$$y'_t = \text{argmax}(P_{\text{dp}}(y_t)) \quad (4)$$

However, although this direct prediction method is effective on the fixed-length grammatical error correction problem, it can only conduct the same-positional substitution operation. For complex correcting cases which require deletion, insertion, and local paraphrasing, the performance is unacceptable. This inferior performance phenomenon is also discussed in the tasks of non-autoregressive neural machine translation (Gu and Kong, 2020).

One of the essential reasons causing the inferior performance is that the dependency information among the neighbour tokens are missed. Therefore, dependency modeling should be called back to improve the performance of generation. Naturally, linear-chain CRF (Lafferty et al., 2001) is introduced to fix this issue, and luckily, Sun et al. (2019) also employ CRF to address the problem of non-autoregressive sequence generation, which inspired us a lot.

Dependency Modeling via CRF Then given the input sequence X , under the CRF framework, the likelihood of the target sequence Y with length T'

is constructed as:

$$P_{\text{crf}}(Y|X) = \frac{1}{Z(X)} \exp \left(\sum_{t=1}^{T'} s(y_t) + \sum_{t=2}^{T'} t(y_{t-1}, y_t) \right) \quad (5)$$

where $Z(X)$ is the normalizing factor and $s(y_t)$ represents the label score of y at position t , which can be obtained from the predicted logit vector $\mathbf{s}_t \in \mathbb{R}^{|\mathcal{V}|}$ from Eq. (3), i.e., $\mathbf{s}_t(\mathcal{V}^{y_t})$, where \mathcal{V}^{y_t} is the vocabulary index of token y_t . The value $t(y_{t-1}, y_t) = \mathbf{M}_{y_{t-1}, y_t}$ denotes the transition score from token y_{t-1} to y_t where $\mathbf{M} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is the transition matrix, which is the core term to conduct dependency modeling. Usually, \mathbf{M} can be learnt as neural network parameters during the end-to-end training procedure. However, $|\mathcal{V}|$ is typically very large especially in the text generation scenarios (more than $32k$), therefore it is infeasible to obtain \mathbf{M} and $Z(X)$ efficiently in practice. To overcome this obstacle, as the method used in (Sun et al., 2019), we introduce two low-rank neural parameter metrics $\mathbf{E}_1, \mathbf{E}_2 \in \mathbb{R}^{|\mathcal{V}| \times d_m}$ to approximate the full-rank transition matrix \mathbf{M} by:

$$\mathbf{M} = \mathbf{E}_1 \mathbf{E}_2^\top \quad (6)$$

where $d_m \ll |\mathcal{V}|$. To compute the normalizing factor $Z(X)$, the original Viterbi algorithm (Forney, 1973; Lafferty et al., 2001) need to search all paths. To improve the efficiency, here we only visit the truncated top- k nodes at each time step approximately (Sun et al., 2019).

2.5 Training with Focal Penalty

Considering the characteristic of the directly bottom-to-up information conveying of the task CGEC, therefore, both tasks, direct prediction and CRF-based dependency modeling, can be incorporated jointly into a unified framework during the training stage. The reasons are that, intuitively, direct prediction will focus on the fine-grained predictions at each position, while CRF-layer will pay more attention to the high-level quality of the whole global sequence. We employ Maximum Likelihood Estimation (MLE) to conduct parameter learning and treat negative log-likelihood (NLL) as the loss function. Thus, the optimization objective for direct prediction \mathcal{L}_{dp} is:

$$\mathcal{L}_{\text{dp}} = - \sum_{t=1}^{T'} \log P_{\text{dp}}(y_t|X) \quad (7)$$

And the loss function \mathcal{L}_{crf} for CRF-based dependency modeling is:

$$\mathcal{L}_{\text{crf}} = - \log P_{\text{crf}}(Y|X) \quad (8)$$

Then the final optimization objective is:

$$\mathcal{L} = \mathcal{L}_{\text{dp}} + \mathcal{L}_{\text{crf}} \quad (9)$$

As mentioned in Section 1, one obvious but crucial phenomenon for CGEC is that most words in a sentence are correct and need not to be changed. Considering that maximum likelihood estimation is used as the parameter learning approach in those two tasks, then a simple copy strategy can lead to a sharp decline in terms of loss functions. Then, intuitively, the grammatical error tokens which need to be correctly fixed in practice, unfortunately, attract less attention during the training procedure. Actually, these tokens, instead, should be regarded as the focal points and contribute more to the optimization objectives. However, no previous works investigate this problem thoroughly on the task of CGEC.

To alleviate this issue, we introduce a useful trick, focal loss (Lin et al., 2020), into our loss functions for direct prediction and CRF:

$$\begin{aligned} \mathcal{L}_{\text{dp}}^{\text{fl}} &= - \sum_{t=1}^{T'} (1 - P_{\text{dp}}(y_t|X))^\gamma \log P_{\text{dp}}(y_t|X) \\ \mathcal{L}_{\text{crf}}^{\text{fl}} &= -(1 - P_{\text{crf}}(Y|X))^\gamma \log P_{\text{crf}}(Y|X) \end{aligned} \quad (10)$$

where γ is a hyperparameter to control the penalty weight. It is obvious that $\mathcal{L}_{\text{dp}}^{\text{fl}}$ is penalized on the token level, while $\mathcal{L}_{\text{crf}}^{\text{fl}}$ is weighted on the sample level and will work in the condition of batch-training. The final optimization objective with focal penalty strategy is:

$$\mathcal{L}^{\text{fl}} = \mathcal{L}_{\text{dp}}^{\text{fl}} + \mathcal{L}_{\text{crf}}^{\text{fl}} \quad (11)$$

2.6 Inference

During the inference stage, for the input source sentence X , we can employ the original $|\mathcal{V}|$ nodes Viterbi algorithm to obtain the target global optimal result. We can also utilize the truncated top- k Viterbi algorithm for high computing efficiency (Sun et al., 2019).

Corpus	#Train	#Dev	#Test	Type
SIGHAN15	2,339	-	1,100	FixLen
HybirdSet	274,039	3,162	3,162	FixLen
TtTSet	539,268	5,662	5,662	VarLen

Table 1: Statistics of the datasets.

3 Experimental Setup

3.1 Settings

The core technical components of our proposed TtT is Transformer (Vaswani et al., 2017) and CRF (Lafferty et al., 2001). The pre-trained Chinese BERT-base model (Devlin et al., 2019) is employed to initialize the model. To approximate the transition matrix in the CRF layer, we set the dimension d of matrices E_1 and E_2 as 32. For the normalizing factor $Z(\mathbf{X})$, we set the predefined beam size k as 64. The hyperparameter γ which is used to weight the focal penalty term is set to 0.5 after parameter tuning. Training batch-size is 100, learning rate is $1e - 5$, dropout rate is 0.1. Adam optimizer (Kingma and Ba, 2015) is used to conduct the parameter learning.

3.2 Datasets

The overall statistic information of the datasets used in our experiments are depicted in Table 1.

SIGHAN15 (Tseng et al., 2015)² This is a benchmark dataset for the evaluation of CGEC and it contains 2,339 samples for training and 1,100 samples for testing. As did in some typical previous works (Wang et al., 2019; Zhang et al., 2020b), we also use the SIGHAN15 testset as the benchmark dataset to evaluate the performance of our models as well as the baseline methods in fixed-length (FixLen) error correction settings.

HybirdSet (Wang et al., 2018)³ It is a newly released dataset constructed according to a prepared confusion set based on the results of ASR (Yu and Deng, 2014) and OCR (Tong and Evans, 1996). This dataset contains about 270k paired samples and it is also a FixLen dataset.

TtTSet Considering that datasets of SIGHAN15 and HybirdSet are all FixLen type datasets, in order to demonstrate the capability of our model TtT on the scenario of Variable-Length (VarLen) CGEC, based on the corpus of HybirdSet, we

²<http://ir.itc.ntnu.edu.tw/lre/sighan8csc.html>

³<https://github.com/wdimmy/Automatic-Corpus-Generation>

build a new VarLen dataset. Specifically, operations of deletion, insertion, and local shuffling are conducted on the original sentences to obtain the incorrect samples. Each operation covers one-third of samples, thus we get about 540k samples finally.

3.3 Comparison Methods

We compare the performance of TtT with several strong baseline methods on both FixLen and VarLen settings.

NTOU employs n-gram language model with a reranking strategy to conduct prediction (Tseng et al., 2015).

NCTU-NTUT also uses CRF to conduct label dependency modeling (Tseng et al., 2015).

HanSpeller++ employs Hidden Markov Model with a reranking strategy to conduct the prediction (Zhang et al., 2015).

Hybrid utilizes LSTM-based seq2seq framework to conduct generation (Wang et al., 2018) and **Confusionset** introduces a copy mechanism into seq2seq framework (Wang et al., 2019).

FASpell incorporates BERT into the seq2seq for better performance (Hong et al., 2019).

SoftMask-BERT firstly conducts error detection using a GRU-based model and then incorporating the predicted results with the BERT model using a soft-masked strategy (Zhang et al., 2020b). Note that the best results of **SoftMask-BERT** are obtained after pre-training on a large-scale dataset with 500M paired samples.

SpellGCN proposes to incorporate phonological and visual similarity knowledge into language models via a specialized graph convolutional network (Cheng et al., 2020).

Chunk proposes a chunk-based decoding method with global optimization to correct single character and multi-character word typos in a unified framework (Bao et al., 2020).

We also implement some classical methods for comparison and ablation analysis, especially for the VarLen correction problem. **Transformer-s2s** is the typical Transformer-based seq2seq framework for sequence prediction (Vaswani et al., 2017). **GPT2-finetune** is also a sequence generation framework fine-tuned based on a pre-trained Chinese GPT2 model⁴ (Radford et al., 2019; Li, 2020). **BERT-finetune** is just fine-tune the Chinese BERT model on the CGEC corpus directly. Beam search decoding strategy is employed to con-

⁴<https://github.com/lipiji/Guyu>

Model	Detection				Correction			
	ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
NTOU (2015)	42.2	42.2	41.8	42.0	39.0	38.1	35.2	36.6
NCTU-NTUT (2015)	60.1	71.7	33.6	45.7	56.4	66.3	26.1	37.5
HanSpeller++ (2015)	70.1	80.3	53.3	64.0	69.2	79.7	51.5	62.5
Hybird (2018)	-	56.6	69.4	62.3	-	-	-	57.1
FASPELL (2019)	74.2	67.6	60.0	63.5	73.7	66.6	59.1	62.6
Confusionset (2019)	-	66.8	73.1	69.8	-	71.5	59.5	64.9
SoftMask-BERT (2020b)	80.9	73.7	73.2	73.5	77.4	66.7	66.2	66.4
Chunk (2020)	76.8	88.1	62.0	72.8	74.6	87.3	57.6	69.4
SpellGCN (2020)	-	74.8	80.7	77.7	-	72.1	77.7	75.9
Transformer-s2s (Sec.3.3)	67.0	73.1	52.2	50.9	66.2	72.5	50.6	59.6
GPT2-finetune (Sec.3.3)	65.1	70.0	51.9	59.4	64.6	69.1	50.7	58.5
BERT-finetune (Sec.3.3)	75.4	84.1	61.5	71.1	71.6	82.2	53.9	65.1
TtT (Sec.2)	82.7	85.4	78.1	81.6	81.5	85.0	75.6	80.0

Table 2: Detection and Correction results evaluated on the SIGHAN2015 testset (1100 samples).

Model	Detection				Correction			
	ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
Transformer-s2s (Sec.3.3)	25.6	65.6	16.1	25.9	24.6	63.6	14.8	24.0
GPT2-finetune (Sec.3.3)	51.3	85.2	47.9	61.3	45.1	82.8	40.2	54.1
BERT-finetune (Sec.3.3)	46.8	89.0	38.9	54.1	36.9	84.8	26.7	40.7
TtT (Sec.2)	55.6	89.8	50.4	64.6	60.6	88.5	44.2	58.9

Table 3: Detection and Correction results evaluated on the TtTSet testset (5662 samples).

duct generation for Transformer-s2s and GPT2-finetune, and beam-size is 5. Note that some of the original methods above mentioned can only work in the FixLen settings, such as **SoftMask-BERT** and **BERT-finetune**.

3.4 Evaluation Metrics

Following the typical previous works (Wang et al., 2019; Hong et al., 2019; Zhang et al., 2020b), we employ sentence-level **Accuracy**, **Precision**, **Recall**, and **F1-Measure** as the automatic metrics to evaluate the performance of all systems⁵. We also report the detailed results for error **Detection** (all locations of incorrect characters in a given sentence should be completely identical with the gold standard) and **Correction** (all locations and corresponding corrections of incorrect characters should be completely identical with the gold standard) respectively (Tseng et al., 2015).

4 Results and Discussions

4.1 Results in FixLen Scenario

Table 2 depicts the main evaluation results of our proposed framework **TtT** as well as the comparison baseline methods. It should be emphasized

⁵<http://nlp.ee.ncu.edu.tw/resource/csc.html>

that SoftMask-BERT is pre-trained on a 500M-size paired dataset. Our model **TtT**, as well as the baseline methods such as Transformer-s2s, GPT2-finetune, BERT-finetune, and Hybird are all trained on the 270k-size HybirdSet. Nevertheless, **TtT** obtains improvements on the tasks of error Detection (F1:77.7 \rightarrow 81.6) and Correction (F1:75.9 \rightarrow 80.0) compared to all strong baselines on F1 metric, which indicates the superiority of our proposed approach.

4.2 Results in VarLen Scenario

Benefit from the CRF-based dependency modeling component, **TtT** can conduct deletion, insertion, local paraphrasing operations jointly to address the Variable-Length (VarLen) error correction problem. The experimental results are described in Table 3. Considering that those sequence generation methods such as Transformer-s2s and GPT2-finetune can also conduct VarLen correction operation, thus we report their results as well. From the results, we can observe that **TtT** can also achieve a superior performance in the VarLen scenario. The reasons are clear: BERT-finetune as well as the related methods are not appropriate in VarLen scenario, especially when the target is longer than the input. The text generation models such as Transformer-s2s and GPT2-finetune suffer from the problem of hallucination (Maynez et al., 2020) and repetition,

TrainSet	Model	Detection				Correction			
		ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
SIGHAN15	Transformer-s2s	46.5	42.2	23.6	30.3	43.4	34.9	17.3	23.2
	GPT2-finetune	45.2	42.3	30.8	35.7	42.6	37.7	25.5	30.4
	BERT-finetune	35.8	34.1	32.8	33.4	31.3	27.1	23.6	25.3
	TtT	51.3	50.6	38.0	43.4	45.8	41.9	26.7	32.7
HybirdSet	Transformer-s2s	67.0	73.1	52.2	50.9	66.2	72.5	50.6	59.6
	GPT2-finetune	65.1	70.0	51.9	59.4	64.6	69.1	50.7	58.5
	BERT-finetune	75.4	84.1	61.5	71.1	71.6	82.2	53.9	65.1
	TtT	82.7	85.4	78.1	81.6	81.5	85.0	75.6	80.0

Table 4: Performance of models trained on different datasets.

TrainSet	Model	Detection				Correction			
		ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
SIGHAN15	TtT w/o \mathcal{L}_{crf}	35.8	34.1	32.8	33.4	31.3	27.1	23.6	25.3
	TtT w/o \mathcal{L}_{dp}	35.5	32.0	28.0	29.9	31.2	24.9	19.3	21.6
	TtT	42.6	39.4	31.5	35.0	36.7	28.9	23.6	26.0
HybirdSet	TtT w/o \mathcal{L}_{crf}	75.4	84.1	61.5	71.1	71.6	82.2	53.9	65.1
	TtT w/o \mathcal{L}_{dp}	81.2	83.4	77.1	80.1	80.0	83.0	74.7	78.6
	TtT	82.7	85.6	77.9	81.5	81.1	85.0	74.7	79.5

Table 5: Ablation analysis of \mathcal{L}_{dp} and \mathcal{L}_{crf} .

TrainSet	γ	Detection				Correction			
		ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
SIGHAN15	0.0	42.6	39.4	31.5	35.0	36.7	28.9	23.6	26.0
	0.1	48.8	47.0	35.5	40.3	43.8	38.7	25.1	30.4
	0.5	51.3	50.6	38.0	43.4	45.8	41.9	26.7	32.6
	1.0	51.8	51.3	37.7	43.5	46.3	42.5	26.5	32.6
	2.0	50.0	48.6	36.3	41.5	44.4	39.5	25.0	30.6
	5.0	48.9	47.1	37.2	47.6	42.8	37.6	25.1	30.6
HybirdSet	0.0	82.7	85.6	77.9	81.5	81.1	85.0	74.7	79.5
	0.1	74.6	73.5	75.4	74.4	73.2	72.7	72.6	72.7
	0.5	82.7	85.4	78.0	81.6	81.5	85.0	75.6	80.0
	1.0	81.1	83.2	77.1	80.0	80.0	82.8	74.9	78.6
	2.0	79.2	80.4	76.2	78.2	78.2	80.0	74.1	76.9
	5.0	80.3	81.6	77.3	79.4	78.7	80.9	74.1	77.4

Table 6: Tuning for focal loss hyperparameter γ .

which are not steady on the problem of CGEC.

4.3 Ablation Analysis

Different Training Dataset Recall that we introduce several groups of training datasets in different scales as depicted in Table 1. It is also very interesting to investigate the performances on different-size datasets. Then we conduct training on those training datasets and report the results still on the SIGHAN2015 testset. The results are shown in Table 4. No matter what scale of the dataset is, TtT always obtains the best performance.

Impact of \mathcal{L}_{dp} and \mathcal{L}_{crf} Table 5 describes the performance of our model TtT and the variants without \mathcal{L}_{dp} (TtT w/o \mathcal{L}_{dp}) and \mathcal{L}_{crf} (TtT w/o \mathcal{L}_{crf}). We can conclude that the fusion of these two tasks, direct prediction and CRF-based dependency mod-

eling, can indeed improve the performance.

Parameter Tuning for Focal Loss The focal loss penalty hyperparameter γ is crucial for the loss function $\mathcal{L} = \mathcal{L}_{dp} + \mathcal{L}_{crf}$ and should be adjusted on the specific tasks (Lin et al., 2020). We conduct grid search for $\gamma \in (0, 0.1, 0.5, 1, 2, 5)$ and the corresponding results are provided in Table 6. Finally, we select $\gamma = 0.5$ for TtT for the CGEC task.

4.4 Computing Efficiency Analysis

Practically, CGEC is an essential and useful task and the techniques can be used in many real applications such as writing assistant, post-processing of ASR and OCR, search engine, etc. Therefore, the time cost efficiency of models is a key point which needs to be taken into account. Table 7 depicts the time cost per sample of our model TtT and

Model	Time (ms)	Speedup
Transformer-s2s	815.40	1x
GPT2-finetune	552.82	1.47x
TtT	39.25	20.77x
BERT-finetune	14.72	55.35x

Table 7: Comparisons of the computing efficiency.

some baseline approaches. The results demonstrate that TtT is a cost-effective method with superior prediction performance and low computing time complexity, and can be deployed online directly.

5 Conclusion

We propose a new framework named tail-to-tail non-autoregressive sequence prediction, which abbreviated as **TtT**, for the problem of CGEC. A BERT based sequence encoder is introduced to conduct bidirectional representation learning. In order to conduct substitution, deletion, insertion, and local paraphrasing simultaneously, a CRF layer is stacked on the up tail to conduct non-autoregressive sequence prediction by modeling the dependencies among neighbour tokens. Low-rank decomposition and a truncated Viterbi algorithm are introduced to accelerate the computations. Focal loss penalty strategy is adopted to alleviate the class imbalance problem considering that most of the tokens in a sentence are not changed. Experimental results on standard datasets demonstrate the effectiveness of TtT in terms of sentence-level Accuracy, Precision, Recall, and F1-Measure on tasks of error Detection and Correction. TtT is of low computing complexity and can be deployed online directly.

In the future, we plan to introduce more lexical analysis knowledge such as word segmentation and fine-grained named entity recognition (Zhang et al., 2020a) to further improve the performance.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. [Neural machine translation by jointly learning to align and translate](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Zuyi Bao, Chen Li, and Rui Wang. 2020. [Chunk-based chinese spelling check with global optimization](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*,

pages 2031–2040. Association for Computational Linguistics.

- Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. [The BEA-2019 shared task on grammatical error correction](#). In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2019, Florence, Italy, August 2, 2019*, pages 52–75. Association for Computational Linguistics.
- Chao-Huang Chang. 1995. [A new approach for automatic chinese spelling correction](#). In *Proceedings of Natural Language Processing Pacific Rim Symposium*, pages 278–283. Citeseer.
- Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. [Spellgc: Incorporating phonological and visual similarities into language models for chinese spelling check](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 871–881. Association for Computational Linguistics.
- Robert Dale, Ilya Anisimoff, and George Narroway. 2012. [HOO 2012: A report on the preposition and determiner error correction shared task](#). In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, BEA@NAACL-HLT 2012, June 7, 2012, Montréal, Canada*, pages 54–62. The Association for Computer Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.
- Huizhong Duan and Bo-June Paul Hsu. 2011. [Online spelling correction for query completion](#). In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 117–126. ACM.
- G David Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.
- Jianfeng Gao, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. 2010. [A large scale ranker-based system for search query spelling correction](#). In *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*, pages 358–366. Tsinghua University Press.
- Tao Ge, Furu Wei, and Ming Zhou. 2018. [Reaching human-level performance in automatic grammatical error correction: An empirical study](#). *CoRR*, abs/1807.01270.

- M Ali Ghufuron and Fathia Rosyida. 2018. [The role of grammarly in assessing english as a foreign language \(efl\) writing](#). *Lingua Cultura*, 12(4):395–403.
- Jiatao Gu and Xiang Kong. 2020. [Fully non-autoregressive neural machine translation: Tricks of the trade](#). *CoRR*, abs/2012.15833.
- Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. [Incorporating copying mechanism in sequence-to-sequence learning](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.
- Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu. 2019. [Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm](#). In *Proceedings of the 5th Workshop on Noisy User-generated Text, W-NUT@EMNLP 2019, Hong Kong, China, November 4, 2019*, pages 160–169. Association for Computational Linguistics.
- Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. [Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4248–4254. Association for Computational Linguistics.
- Clare-Marie Karat, Christine Halverson, Daniel B. Horn, and John Karat. 1999. [Patterns of entry and correction in large vocabulary continuous speech recognition system](#). In *Proceeding of the CHI '99 Conference on Human Factors in Computing Systems: The CHI is the Limit, Pittsburgh, PA, USA, May 15-20, 1999*, pages 568–575. ACM.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Marek Kubis, Zygmunt Vetulani, Mikolaj Wypych, and Tomasz Zietkiewicz. 2020. [Open challenge for correcting errors of speech recognition systems](#). *CoRR*, abs/2001.03041.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. [Conditional random fields: Probabilistic models for segmenting and labeling sequence data](#). In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.
- Piji Li. 2020. [An empirical investigation of pre-trained transformer language models for open-domain dialogue generation](#). *CoRR*, abs/2003.04195.
- Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, and Jinpeng Dong. 2020. [BERT enhanced neural machine translation and sequence tagging model for Chinese grammatical error diagnosis](#). In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 57–66, Suzhou, China. Association for Computational Linguistics.
- Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2020. [Focal loss for dense object detection](#). *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(2):318–327.
- Bruno Martins and Mário J. Silva. 2004. [Spelling correction for search engine queries](#). In *Advances in Natural Language Processing, 4th International Conference, EsTAL 2004, Alicante, Spain, October 20-22, 2004, Proceedings*, volume 3230 of *Lecture Notes in Computer Science*, pages 372–383. Springer.
- Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan T. McDonald. 2020. [On faithfulness and factuality in abstractive summarization](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1906–1919. Association for Computational Linguistics.
- In Jae Myung. 2003. [Tutorial on maximum likelihood estimation](#). *Journal of mathematical Psychology*, 47(1):90–100.
- Courtney Napoles, Keisuke Sakaguchi, and Joel R. Tetreault. 2017. [JFLEG: A fluency corpus and benchmark for grammatical error correction](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 229–234. Association for Computational Linguistics.
- Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. [The conll-2014 shared task on grammatical error correction](#). In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2014, Baltimore, Maryland, USA, June 26-27, 2014*, pages 1–14. ACL.
- Feng Nie, Jin-Ge Yao, Jinpeng Wang, Rong Pan, and Chin-Yew Lin. 2019. [A simple recipe towards reducing hallucination in neural surface realisation](#). In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2673–2679. Association for Computational Linguistics.
- Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem N. Chernodub, and Oleksandr Skurzhanskyi. 2020. [Gector - grammatical error correction: Tag, not](#)

- rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2020, Online, July 10, 2020*, pages 163–170. Association for Computational Linguistics.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. [Sequence level training with recurrent neural networks](#). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.
- Alla Rozovskaya, Houda Bouamor, Nizar Habash, Wajdi Zaghoulani, Ossama Obeid, and Behrang Mohit. 2015. [The second QALB shared task on automatic text correction for arabic](#). In *Proceedings of the Second Workshop on Arabic Natural Language Processing, ANLP@ACL 2015, Beijing, China, July 30, 2015*, pages 26–35. Association for Computational Linguistics.
- Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhi-Hong Deng. 2019. [Fast structured decoding for sequence models](#). In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3011–3020.
- Xiang Tong and David A. Evans. 1996. [A statistical approach to automatic OCR error correction in context](#). In *Fourth Workshop on Very Large Corpora, VLC@COLING 1996, Copenhagen, Denmark, August 4, 1996*.
- Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. [Introduction to SIGHAN 2015 bake-off for chinese spelling check](#). In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing, SIGHAN@IJCNLP 2015, Beijing, China, July 30-31, 2015*, pages 32–37. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.
- Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. [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, October 31 - November 4, 2018*, pages 2517–2527. Association for Computational Linguistics.
- Dingmin Wang, Yi Tay, and Li Zhong. 2019. [Confusionset-guided pointer networks for chinese spelling check](#). In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5780–5785. Association for Computational Linguistics.
- Haoyu Wang, Shuyan Dong, Yue Liu, James Logan, Ashish Kumar Agrawal, and Yang Liu. 2020a. [ASR error correction with augmented transformer for entity retrieval](#). In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 1550–1554. ISCA.
- Hongfei Wang, Michiki Kurosawa, Satoru Katsumata, and Mamoru Komachi. 2020b. [Chinese grammatical correction using bert-based pre-trained model](#). In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*, pages 163–168. Association for Computational Linguistics.
- Yu Wang, Yuelin Wang, Jie Liu, and Zhuo Liu. 2020c. [A comprehensive survey of grammar error correction](#). *CoRR*, abs/2005.06600.
- Junwen Xing, Longyue Wang, Derek F. Wong, Lidia S. Chao, and Xiaodong Zeng. 2013. [Um-checker: A hybrid system for english grammatical error correction](#). In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 34–42. ACL.
- Dong Yu and Li Deng. 2014. *Automatic Speech Recognition: A Deep Learning Approach*. Springer.
- Haisong Zhang, Lemaoy Liu, Haiyun Jiang, Yangming Li, Enbo Zhao, Kun Xu, Linfeng Song, Suncong Zheng, Botong Zhou, Jianchen Zhu, Xiao Feng, Tao Chen, Tao Yang, Dong Yu, Feng Zhang, Zhanhui Kang, and Shuming Shi. 2020a. [Textsmart: A text understanding system for fine-grained NER and enhanced semantic analysis](#). *CoRR*, abs/2012.15639.
- Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020b. [Spelling error correction with soft-masked BERT](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 882–890. Association for Computational Linguistics.
- Shuiyuan Zhang, Jinhua Xiong, Jianpeng Hou, Qiao Zhang, and Xueqi Cheng. 2015. [Hanspeller++: A unified framework for chinese spelling correction](#). In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing, SIGHAN@IJCNLP 2015, Beijing, China, July 30-31, 2015*, pages 38–45. Association for Computational Linguistics.

- Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. [Bridging the gap between training and inference for neural machine translation](#). In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4334–4343. Association for Computational Linguistics.
- Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. [Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 156–165. Association for Computational Linguistics.
- Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. [Overview of the NLPCC 2018 shared task: Grammatical error correction](#). In *Natural Language Processing and Chinese Computing - 7th CCF International Conference, NLPCC 2018, Hohhot, China, August 26-30, 2018, Proceedings, Part II*, volume 11109 of *Lecture Notes in Computer Science*, pages 439–445. Springer.