CRASpell: A Contextual Typo Robust Approach to Improve Chinese Spelling Correction

Shulin Liu[†], **Shengkang Song**[†,‡], **Tianchi Yue**[†],
Tao Yang[†], **Huihui Cai**[†], **Tinghao Yu**[†], **Shengli Sun**[‡]
[†]Tencent AI Platform Department, China
[‡]Peking University, China
{forestliu,tianchiyue}@tencent.com
songsk@stu.pku.edu.cn,slsun@pku.edu.cn

Abstract

Chinese spelling correction (CSC) models detect and correct a typo in texts based on the misspelled character and its context. Recently, Bert-based models have dominated the research of Chinese spelling correction (CSC). These methods have two limitations: (1) they have poor performance on multi-typo texts. In such texts, the context of each typo contains at least one misspelled character, which brings noise information. Such noisy context leads to the declining performance on multi-typo texts. (2) they tend to overcorrect valid expressions to more frequent expressions due to the masked token recovering task of Bert. We attempt to address these limitations in this paper. To make our model robust to contextual noise brought by typos, our approach first constructs a noisy context for each training sample. Then the correction model is forced to yield similar outputs based on the noisy and original contexts. Moreover, to address the overcorrection problem, copy mechanism is incorporated to encourage our model to prefer to choose the input character when the miscorrected and input character are both valid according to the given context. Experiments are conducted on widely used benchmarks. Our model achieves superior performance against state-of-the-art methods by a remarkable gain. We release the source code and pre-trained model for further use by the community[1].

1 Introduction

Chinese spelling correction (CSC) is a task to detect and correct spelling errors in texts(Chen et al., 2013; Yu and Li, 2014). It is a challenging yet important task, which plays an important role in various NLP applications such as optical character recognition(Afli et al., 2016; Dong and Smith, 2018), automatic speech recognition(Sarma and Palmer, 2004; Errattahi et al., 2018) and search engine(Martins and Silva, 2004; Gao et al., 2010).

[1]https://github.com/liushulinle/CRASpell

ID	Text	Correction
E1	我是你得(de)学生。	的(de)
E2	他很少座(zuo)捷运粗(cu)去玩。	坐(zuo),出(chu)

Table 1: Examples of Chinese spelling errors, where error characters are marked in red and their phonics are given in brackets.

Corpus	Sentence-level			Character-level		
	me	te	$ratio$	me	te	$ratio$
S13	200	969	21%	450	1,219	37%
S14	155	535	29%	406	766	53%
S15	121	550	22%	284	704	40%

Table 2: The multi-typo statistics of **SIGHAN13**, **SIGHAN14** and **SIGHAN15**, where me denotes the number of multi-typo sentences, te denotes the number of total misspelled sentences, $ratio = \frac{me}{te}$. The character-level me denotes the amount of misspelled characters in multi-typo sentences.

In Chinese, spelling error is mainly caused by the misuse of phonologically and visually similar characters(Liu et al., 2021). According to Liu et al. (2010), about 83% of errors are related to phonological similarity and 48% are related to visual similarity. Table1 illustrates two examples.

In recent years, BERT(Devlin et al., 2019) based models have dominated the research of Chinese spelling correction(Cheng et al., 2020; Zhang et al., 2020; Bao et al., 2020; Liu et al., 2021; Li and Shi, 2021; Huang et al., 2021), which follow the paradigm of non-autoregressive generation. Typically, these models generate corrected characters for all input characters in parallel, where the generated characters can be the same as the input characters. Thanks to the success of BERT, these models significantly improved the results of CSC benchmarks. However, they have two limitations.

First, multi-typo texts are very common in CSC datasets, which is defined as texts containing more than one typos. Table 2 presents

Findings of the Association for Computational Linguistics: ACL 2022, pages 3008 - 3018

Model	Whole Set			Multi-typo Set		
	P	R	F	P	R	F
BERT	97.0	79.3	87.3	96.4	67.4	79.3
Zhang et al. (2020)	96.9	82.9	89.4	91.2	73.8	81.6
Cheng et al. (2020)	96.7	81.4	88.4	95.9	69.9	80.9
Liu et al. (2021)	97.2	85.0	90.7	94.0	75.9	84.0

Table 3: The correction performance of various CSC models on the whole and multi-typo evaluation set.

the multi-typo statistics of SIGHAN13(Wu et al., 2013), SIGHAN14(Yu et al., 2014) and SIGHAN15(Tseng et al., 2015). However, we find the performance of existing CSC models declines sharply on multi-typo texts. Table 3 illustrates the results of the latest CSC models on SIGHAN15 and a multi-typo subset extracted from it. We observe that all models in this table perform worse on multi-typo set than on the whole set. Generally, CSC models detect and correct a typo based on the misspelled character and its context. In multi-typo texts, the context of each character contains at least one typo, which results in noisy information. Take *E2* in Table 1 as an example, the context of "座" contains the typo "粗" and vice versa. Such noisy context leads to the declining performance on multi-typo texts. We call this problem *Contextual Typo Disturbance*.

Second, Bert is a masked language model(Devlin et al., 2019). It learned how to recover a masked token based on its context from large corpus. When there are multiple valid characters for a masked position, the model prefers to recover it with the most frequent one in the training corpus. As a consequence, Bert-based models tend to overcorrect an infrequent but valid expression to a more frequent expression. For instance, both BERT(Cheng et al., 2020; Liu et al., 2021) and PLOME(Liu et al., 2021) wrongly correct "这并非是说..." to "这并不是说...". We call this problem *Overcorrection*.

In this paper, we attempt to address the aforementioned problems by proposing a new model called CRASpell, which is short for **C**ontextual typo **R**obust **A**pproach for Chinese spelling correction. Figure 1 illustrates the framework. The *Contextual Typo Disturbance* problem is caused by the noise of contextual typos, therefore we try to make our model robust to such noise. To this end, our approach first generates a noisy context for each training instance, and then forces the correction model to yield similar outputs based on the original and noisy context. Moreover, to address the *Overcorrection* problem, we incorporate the copy mechanism(Gu et al., 2016; Zeng et al., 2018) in our model. Finally, the output for each position in a given text is the sum of generative distribution and copy distribution. Thus, our model has more chances to keep the input character unchanged when the miscorrected and input characters are both valid according to the given context.

We conduct experiments on the widely used benchmark dataset SIGHAN(Wu et al., 2013; Yu et al., 2014; Tseng et al., 2015). The experimental results show that our model outperforms all the compared approaches. Furthermore, we extract samples containing multiple typos from SIGHAN to construct a multi-typo evaluation set. Experimental results show that our approach achieves 2.6% absolute improvement in detection and 2.7% absolute improvement in correction on the multi-typo set, which demonstrates the effectiveness of our approach for multi-typo texts.

We summarize our contributions as follows. (1) we point out two limitations of existing CSC approaches, which are called the *Contextual Typo Disturbance* problem and the *Overcorrection* problem; (2) we propose an effective model to address these limitations; (3) our model outperforms state-of-the-art methods with remarkable gains.

2 Related Work

Chinese spelling correction is a challenging task in natural language processing, which plays important roles in many applications, such as search engine (Martins and Silva, 2004; Gao et al., 2010), automatic essay scoring (Burstein and Chodorow, 1999; Lonsdale and Strong-Krause, 2003), and optical character recognition (Afli et al., 2016; Wang et al., 2018). It has been an active topic, and various approaches have been proposed in recent years (Yu and Li, 2014; Wang et al., 2018, 2019; Zhang et al., 2020; Cheng et al., 2020; Liu et al., 2021; Li and Shi, 2021; Huang et al., 2021).

Early work on CSC followed the pipeline of error identification, candidate generation and selection. Some researchers focused on unsupervised approaches, which typically adopted a confusion set to find correct candidates and employed language model to select the correct one (Chang, 1995; Huang et al., 2000; Chen et al., 2013; Yu and Li, 2014; Tseng et al., 2015). However, these methods failed to condition the correction on the input

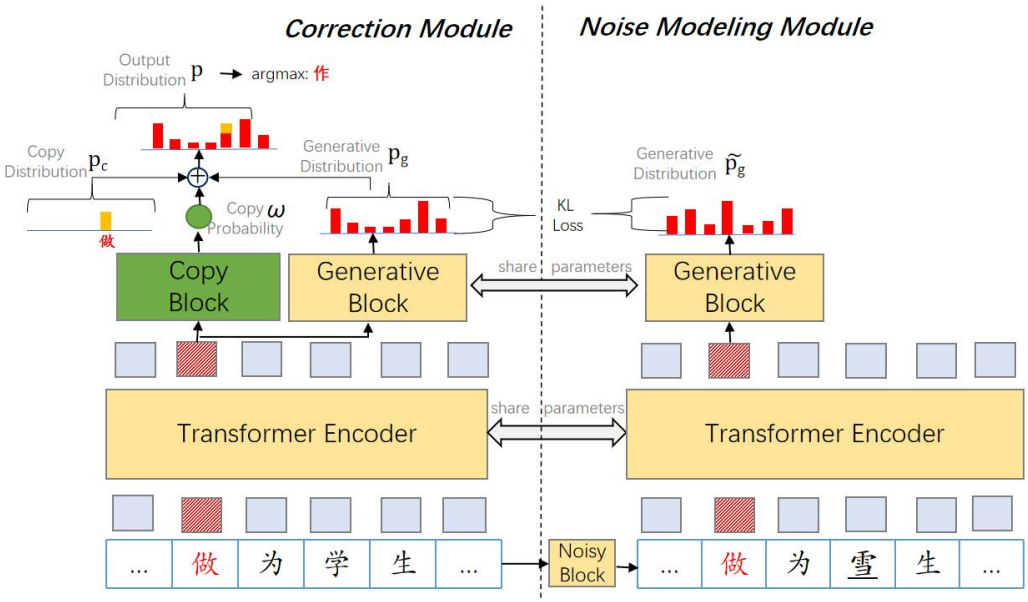

Figure 1: Framework of the proposed *CRASpell*. **Left**: This component presents the correction module, which illustrates how our model makes prediction for the red character "做". **Right**: This component presents the noise modeling module, which forces the model to yield similar distribution for "做" separately based on the original and noisy contexts. This module is only active in the training process.

sentence. In order to model the input context, discriminative sequence tagging methods (Wang et al., 2018) and sequence-to-sequence generative models (Chollampatt et al., 2016; Ji et al., 2017; Ge et al., 2018; Wang et al., 2019) were employed.

BERT (Devlin et al., 2019) is a bidirectional language model based on Transformer encoder (Vaswani et al., 2017). It has been demonstrated effective in a wide range of applications, such as question answering (Yang et al., 2019), information extraction (Lin et al., 2019), and semantic matching (Reimers and Gurevych, 2019). Recently, it has dominated the researches on CSC (Hong et al., 2019; Zhang et al., 2020; Cheng et al., 2020; Liu et al., 2021; Li and Shi, 2021; Huang et al., 2021). Hong et al. (2019) adopted the DAE-Decoder paradigm with BERT as encoder. Zhang et al. (2020) introduced a detection network to generate the masking vector for the BERT-based correction network. Cheng et al. (2020) employed graph convolution network (GCN) (Kipf and Welling, 2016) combined with BERT to model character inter-dependence. In our previous work (Liu et al., 2021), we proposed a task-specific pretrained model for Chinese spelling correction, which masked tokens by similar characters according to confusion set. Li and Shi (2021) proposed to decode with the CRF module. Huang et al. (2021) proposed to incorporate phonological and visual features via a multi-modal module. Although these models achieved some success on benchmarks, all of them suffer from the typo disturbance problem and overcorrection problem.

3 Approach

In this section, we describe the proposed approach and its detailed implementation. Figure 1 illustrates the framework of our model, which is composed of correction module and noise modeling module.

3.1 Task Formulation

Chinese spelling correction aims to detect and correct spelling errors in texts. Recent BERT-based approaches(Cheng et al., 2020; Zhang et al., 2020; Liu et al., 2021; Li and Shi, 2021; Huang et al., 2021) modeled it as a non-autoregressive generation task. Formally, given a character sequence $\mathbf{X} = \{x_1, x_2, ..., x_n\}$ consisting of n characters, the model is expected to generate a target sequence $\mathbf{Y} = \{y_1, y_2, ..., y_n\}$ with the same length as that of $\mathbf{X}$, where typos in $\mathbf{X}$ are corrected in $\mathbf{Y}$.

3.2 Correction Module

As illustrated in Figure 1(left part), the input of this module is the sequence of embeddings $\mathbf{E} = \{\mathbf{e_1}, \mathbf{e_2}, ..., \mathbf{e_n}\}$, where $\mathbf{e_i}$ denotes the embedding of character x_i in a given text $\mathbf{X} = \{x_1, x_2, ..., x_n\}$, which is the sum of word embedding, position em-

bedding and segment embedding of the character. Then, $\mathbf{E}$ is fed into the transformer encoder, which has the same architecture and configuration as that of Bert$_{base}$(Devlin et al., 2019). The transformer encoder generates hidden representation matrix $\mathbf{H} = \{\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_n}\}$ for $\mathbf{X}$, where $\mathbf{h_i} \in \mathbf{R}^{768}$ is the representation of x_i. The final output of this module is the weighted sum of generative distribution and copy distribution, where the weight is the copy probability learned by the model.

Generative Distribution The generative distribution, $\mathbf{p_g} \in \mathbf{R}^{n_v}$, is computed by the generative block in Figure 1, which is an one-layer feed forward network with softmax normalization. The following equation describes how this block works:

$$\mathbf{p_g} = softmax(\mathbf{W_g h_i} + \mathbf{b_g}) \tag{1}$$

where $\mathbf{W_g} \in \mathbf{R}^{n_v \times 768}$ and $\mathbf{b_g} \in \mathbf{R}^{768}$ are generative parameters, n_v is the size of vocabulary.

Copy Distribution Denote the index of x_i in the vocabulary as $idx(x_i)$, then the copy distribution of x_i, $\mathbf{p_c} \in \{0, 1\}^{n_v}$, is a one-hot vector satisfying:

$$\mathbf{p_c}[k] = \begin{cases} 0 & k \neq idx(x_i) \\ 1 & k = idx(x_i) \end{cases} \tag{2}$$

Copy Probability The copy probability, $\omega \in \mathbf{R}$, is computed by the copy block in Figure 1, which is a two-layer feed forward network with layer normalization. The following equations show how this block works:

$$\begin{aligned} \mathbf{h_c} &= \mathbf{W_{ch}} f_{ln}(\mathbf{h_i}) + \mathbf{b_{ch}} \\ \mathbf{h_c'} &= f_{ln}(f_{act}(\mathbf{h_c})) \\ \omega &= Sigmoid(\mathbf{W_c h_c'}) \end{aligned} \tag{3}$$

where $\mathbf{h_i}$ is the hidden representation of x_i generated by the transformer block, $\mathbf{W_{ch}} \in \mathbf{R}^{768 \times d_c}$, $\mathbf{b_{ch}} \in \mathbf{R}^{d_c}$, $\mathbf{W_c} \in \mathbf{R}^{d_c \times 1}$ are model parameters, f_{act} is the activation function, f_{ln} is the layer normalization function.

The final output distribution, $\mathbf{p}$, is computed by the following equation:

$$\mathbf{p} = \omega \times \mathbf{p_c} + (1 - \omega) \times \mathbf{p_g} \tag{4}$$

In previous CSC models(Cheng et al., 2020; Zhang et al., 2020; Liu et al., 2021), the generative distribution $\mathbf{p_g}$ is the final output. These methods suffer the overcorrection problem due to the masked token recovering task of BERT. On contrast, our model incorporates the copy distribution $\mathbf{p_c}$ in the final output, which enables our model to have more chances to choose the input character when it is valid but not the best for BERT.

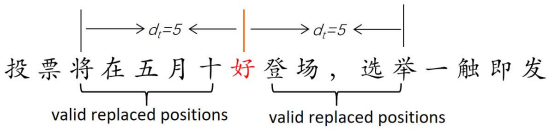

Figure 2: The illustration of valid replaced positions when $d_t = 5$, where the red character is a typo.

3.3 Noise Modeling Module

The noise modeling module is designed to solve the contextual typo disturbance problem by encouraging the correction model to yield similar distributions for the original and noisy contexts. As illustrated in Figure 1 (right part), this module first generates a noisy context based on the input sample, then takes the noisy context as input and yields a generative distribution. At last, the generative distribution is forced to be similar with that generated by the correction module. Note that the noise modeling module is only active in training process.

Noisy Block This component generates noisy contexts by replacing characters of the original training samples. There are two factors affect the quality of the generated noisy context, which can be represented by the following questions.

- Which positions should be replaced?
  Table 3 shows that contextual typo significantly declines the recall score on multi-typo set. We believe this phenomenon occurs because the noise around typos affects CSC models' correction of typos. Therefore, we sample from typo-around positions for replacing, which is defined as positions less than d_t tokens away from the nearest typo. Figure 2 illustrates the valid replaced positions when $d_t = 5$. If a text does not contain any typo, the noisy block will directly output the original text without replacing any positions.

  In our work, we replace at most one position for each typo. If there already exists a typo in the valid positions, we will not replace any position. Our preliminary experiments show that more replaced positions declines the performance. We believe the reason is that too many replaced positions results in too much noise in the context, which declines the performance on non multi-typo texts.

- What characters should be replaced with?
  Following our previous work (Liu et al., 2021), to simulate true contextual typos, we replace

each chosen position by a similar character according to a publicly available confusion set(Wu et al., 2013). Specifically, we replace a chosen position with (i) a random phonologically similar character 70% of the time (ii) a random visually similar character 15% of the time (iii) a random token in the vocabulary 15% of the time.

Generative Distribution Given a training sample $\mathbf{X}$, the noise modeling module first constructs a noisy instance $\widetilde{\mathbf{X}}$ via the noisy block, then yields a generative distribution, $\widetilde{\mathbf{p_g}}$, according to Equation 1 based on $\widetilde{\mathbf{X}}$. Both the transformer encoder and generative block in this module share parameters with that in the correction module.

KL-diverence Loss We force the correction module and noise modeling module to yield similar outputs by minimizing the bidirectional Kullback-Leibler divergence between the generative distributions (see Equation 5).

$$\mathcal{L}_{KL} = \frac{1}{2}(\mathcal{D}_{KL}(\mathbf{p_g}||\widetilde{\mathbf{p_g}}) + \mathcal{D}_{KL}(\widetilde{\mathbf{p_g}}||\mathbf{p_g})) \quad (5)$$

3.4 Learning

Given a training sample ($\mathbf{X}$, $\mathbf{Y}$), the correction loss of the i-th token is defined as:

$$\mathcal{L}_c^i = -\log \mathbf{p}(\mathbf{Y_i}|\mathbf{X}) \quad (6)$$

where $\mathbf{X}$ is a character sequence, $\mathbf{Y}$ is the corrected sequence of $\mathbf{X}$, $\mathbf{p}$ is the output distribution defined in Equation 4. The learning process is driven by optimizing two objectives:

$$\mathcal{L}^i = (1-\alpha_i)\mathcal{L}_c^i + \alpha_i \mathcal{L}_{KL}^i \quad (7)$$

$$\alpha_i = \begin{cases} \alpha, & \widetilde{\mathbf{X}}_i = \mathbf{X}_i \\ 0, & otherwise \end{cases} \quad (8)$$

where α is a trade-off factor for $\mathcal{L}_c$ and $\mathcal{L}_{KL}$. Note that the constructed noise itself will not be involved in the training process but only active as context (see Equation 8). This strategy is designed to ensure that the constructed noise will not change the ratio of positive and negative samples in the training corpus.

4 Experiments

4.1 Datasets

Following previous work(Cheng et al., 2020; Liu et al., 2021; Li and Shi, 2021), the training data is composed of 10K manually annotated samples from SIGHAN(Wu et al., 2013; Yu et al., 2014; Tseng et al., 2015), and 271K automatically generated samples from Wang et al. (2018). To evaluate the performance of the proposed method, we use the test set from the latest SIGHAN benchmark(Tseng et al., 2015) as in (Zhang et al., 2020; Li and Shi, 2021; Liu et al., 2021). This set contains 550 positive samples and 550 negative samples with 461 types of errors, where negative samples denote texts without any typos.

Dataset	*TotalSen*	*TypoSen*	*TypoChar*
SIGHAN15	1100	550	704
MultiTypo	242	121	284

Table 4: The statistics of evaluation datasets, where *TotalSen* is the amount of texts, *TypoSen* is the amount of texts containing typos, *TypoChar* is the amount of misspelled characters. MultiTypo is a subset of SIGHAN15.

Besides, we also construct a multi-typo test set by extracting all the samples containing multiple typos from SIGHAN. To make the ratio of positive and negative samples equal to that of SIGHAN, negative samples are also randomly sampled. Table 4 illustrates the statistics of SIGHAN and multi-typo test set.

4.2 Evaluation Metrics

The most widely used metrics in previous work are precision, recall and F1 scores. However, these metrics were calculated via different methods, which could be grouped into three groups: character-level scores(Wang et al., 2018, 2019; Cheng et al., 2020; Liu et al., 2021), sentence-level scores evaluated based the method from (Hong et al., 2019)[2](Hong et al., 2019; Cheng et al., 2020; Liu et al., 2021) and sentence-level scores based on the method from SighanHan Tools[3](Li and Shi, 2021; Huang et al., 2021).

In this work we choose the character-level scores for the following reasons. (1) Character is the minimum evaluation unit of CSC, thus character-level metrics can reflect the ability of a model in finer gained. (2) The CSC test corpus only contains about 1,000 sentences, but contains tens of thousands of characters. Therefore the results on character level are statistically more confident.

[2]https://github.com/iqiyi/FASPell
[3]http://nlp.ee.ncu.edu.tw/resource/csc.html

4.3 Hyper Parameter Settings

Following previous work(Cheng et al., 2020; Liu et al., 2021), we set the maximum sentence length to 180, batch size to 32 and the learning rate to 5e-5. The hidden size d_c in the copy block is set to 384. The window size d_t for sampling replaced positions is set to 5. The trade-off factor α in Equation 7 is set to 0.05. We initialize the transformer encoder by cBERT released in our previous work (Liu et al., 2021)[4], which has the same architecture with Bert_{base}(Devlin et al., 2019) but is pretrained with misspelled knowledge. Following (Cheng et al., 2020; Liu et al., 2021), all experiments are conducted for 4 runs with different seeds and the averaged metrics are reported.

4.4 Baseline Models

Recent researches have demonstrated that Bert-based models(Zhang et al., 2020; Cheng et al., 2020; Liu et al., 2021; Li and Shi, 2021; Huang et al., 2021) significantly outperform other models including LM-based models(Huang et al., 2000; Chen et al., 2013; Yu and Li, 2014) and non-Bert neural models(Wang et al., 2018, 2019). Therefore, we only compare with recent Bert-based methods. However, it is challenging to compare to all of them since they employed different evaluation methods as mentioned in Section 4.2. We compare with methods who either reported character-level results or released source codes.

- *SoftMask*(Zhang et al., 2020) introduced the soft-masking strategy in Bert to improve the performance of error detection.
- *SpellGCN*(Cheng et al., 2020) combined GCN network with BERT to model the relationship between characters.
- *Tail2Tail*(Li and Shi, 2021) applied the CRF decoder based on BERT.
- *cBERT* is a task-specific pretrained model for CSC proposed in our previous work (Liu et al., 2021), which has the same architecture with Bert but is pretrained with misspelled knowledge. Our model is initialized by cBERT.
- *PLOME*(Liu et al., 2021) is similar with cBERT but incorporates phonological and visual features based on the sequences of phonics and strokes.

As illustrated in Table 5, we also present the performances of these methods on multi-typo subset. All the results are obtained by running publicly available codes, which are SoftMaskBert[5], SpellGCN[6], Tail2Tail[7] and PLOME[8]. Besides, we also implement two baselines:

- *cBERTCopy* employs the copy mechanism (see Section 3.2) based on *cBERT*.
- *cBERTNoise* employs the noise modeling module (see Section 3.3) based on *cBERT*.

Moreover, our noise modeling loss is similar with that in Rdrop(Liang et al., 2021), which fed each sentence into the model twice with drop out operation and encouraged the model to yield similar outputs. To make comparison with Rdrop, we implement it based on cBERT, which is denoted by *cBERTRdrop*.

4.5 Main Results

Table 5 illustrates the performance of the proposed method and baseline models. From the table we observe that:

1) With the incorporation of copy mechanism, *cBERTCopy* achieves consistent improvements against *cBERT* on the precision of all evaluations. This result demonstrates that the copy mechanism can alleviate the overcorrection problem.

2) With the incorporation of noise modeling module, *cBERTNoise* outperforms *cBERT* on all metrics. Especially, *cBERTNoise* significantly improves the detection and correction score by 1.6% and 2.0% on the multi-typo dataset. This result demonstrates that the noise modeling module is very effective for multi-typo texts.

3) *cBERTRdrop* does not replace any positions in the noisy block. We observe that it fails to achieve improvements on the multi-typo set. This result indicates that noisy contexts are necessary to train an effective model for multi-typo texts.

4) The proposed *CRASpell* jointly incorporates the copy mechanism and noise modeling module. It achieves the best performance, indicating that the copy mechanism and noise modeling module are complementary to each other. Moreover, the proposed model outperforms all previous work on both datasets, especially with remarkable gains on the

[4] https://github.com/liushulinle/PLOME

[5] https://github.com/hiyoung123/SoftMaskedBert

[6] https://github.com/ACL2020SpellGCN/SpellGCN

[7] https://github.com/lipiji/TtT

[8] https://github.com/liushulinle/PLOME

Method	Whole Set (1,100 sentences)						Multi-typo Subset (242 sentences)					
	Detection-level			Correction-level			Detection-level			Correction-level		
	P	R	F	P	R	F	P	R	F	P	R	F
SoftMask(Zhang et al., 2020)	75.5*	84.1*	79.6*	96.7*	81.4*	88.4*	86.0	72.9	78.9	95.9	69.9	80.9
SpellGCN(Cheng et al., 2020)	77.7	85.6	81.4	96.9	82.9	89.4	88.8	77.0	82.5	91.2	73.8	81.6
Tail2Tail(Li and Shi, 2021)	75.6*	82.4*	78.9*	96.6*	79.6*	87.3*	86.3	72.0	78.5	94.7	70.8	81.0
PLOME(Liu et al., 2021)	**85.2**	86.8	86.0	**97.2**	85.0	90.7	90.2	80.7	85.2	94.0	75.9	84.0
cBERT(Liu et al., 2021)	83.0	87.8	85.3	96.0	83.9	89.5	90.0	80.3	84.8	94.2	75.6	83.9
cBERTCopy (ours)	84.0	87.7	85.6	96.8	84.8	90.4	90.7	80.2	85.1	95.0	76.1	84.5
cBERTNoise (ours)	83.2	**89.3**	86.1	96.4	86.1	90.9	90.2	83.0	86.4	94.4	78.9	85.9
cBERTRdrop (ours)	83.9	87.8	85.8	96.3	84.6	90.1	91.1	80.6	85.6	93.8	75.6	83.7
CRASpell (ours)	83.5	89.2	**86.3**	97.1	**86.6**	**91.5**	**91.7**	**83.5**	**87.4**	**95.2**	**79.4**	**86.6**

Table 5: The performance of our approach and baseline models on SIGHAN15. Following (Cheng et al., 2020; Liu et al., 2021), we run the experiments 4 times and report the average metrics. All the results on multi-typo subset and results with '*' are obtained by our evaluations.

multi-typo subset. This result further demonstrates that our model is effective to solve the contextual typo disturbance problem.

To make more comprehensive comparisons, we also evaluate the proposed model on SIGHAN14(Yu et al., 2014). Similar with SIGHAN15, we construct a multi-typo test set by extracting a subset from SIGHAN14. Table 6 illustrates the result, from which we observe that *CRASpell* consistently achieves remarkable improvements.

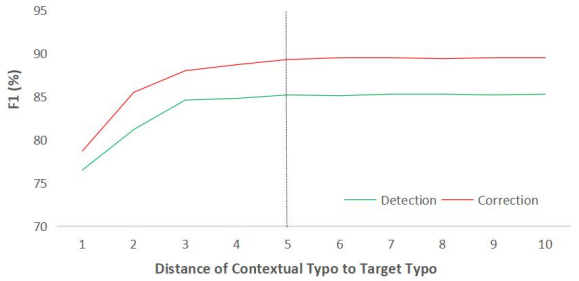

Figure 3: The effects of the contextual typo with different distance to target typos.

4.6 Effects of Different Replaced Positions

In the noisy block (see Section 3.3), we sample typo-around positions for replacing. In this subsection, we implement a new sampling strategy called *Random* for comparison, which randomly samples positions from the whole text for replacing. Table 7 presents the result. We observe that both replacing strategies improve the performance, which demonstrates the effectiveness of the proposed noise modeling module. Furthermore, *Typo-around* strategy outperforms *Random* strategy, verifying our hypothesis in Section 3.3 that previous CSC models performed poorly on multi-typo texts because of the noise around typos.

Moreover, we also investigate the effects of contextual typos with different distances to target typos. To this end, we construct different test sets by adding a contextual typo via randomly replacing a character with different distances to each typo in the SIGHAN15 test set. Then we run *cBERT* on these test sets. Figure 3 illustrates the results, from which we observe that closer contextual typos cause more decrease. Furthermore, when the distance is more than 5, contextual typos nearly have no effects to the CSC model. Therefore, in our experiments we set d_t to 5.

4.7 Effects of Different Replaced Characters

In the noisy block, each chosen position is replaced by characters based on confusion set. In this subsection, we implement a new replacing strategy called *Random* for comparison, which replaces each chosen position by a random character from the vocabulary. Table 8 presents the results. We observe that the *Random* strategy failed to achieve obvious improvements on the multi-typo set. The main reason is that contextual typos constructed by this strategy is significantly different from true contextual typos. On contrast, the *ConfusionSet* strategy achieves improvements with remarkable gains.

4.8 Effects of the Copy Block

Bert(Devlin et al., 2019) is a masked language model and learned lots of common expressions

Method	Whole Set (1,062 sentences)						Multi-typo Subset (310 sentences)					
	Detection-level			Correction-level			Detection-level			Correction-level		
	P	R	F	P	R	F	P	R	F	P	R	F
PLOME(Liu et al., 2021)	77.4	79.6	78.5	**98.8**	78.8	87.7	83.9	72.9	78.0	**99.4**	72.3	83.7
cBERT(Liu et al., 2021)	77.1	79.5	78.3	**98.8**	78.5	87.5	83.5	73.1	77.9	99.3	72.6	83.9
CRASpell(ours)	**78.2**	**82.1**	**80.1**	98.4	**80.8**	**88.7**	**85.8**	**76.8**	**81.1**	98.4	**75.6**	**85.5**

Table 6: The results of our model and baseline models on SIGHAN14. Liu et al. (2021) reported their results on positive samples. The results in this table is obtained by running their code on the whole test set.

Model	Replacing Positions	Whole Set		Multi-typo Set	
		D-F	C-F	D-F	C-F
cBERT	-	85.3	89.5	84.8	83.9
cBERTNoise	Random	85.9	89.9	85.3	85.0
	Typo-around	**86.1**	**90.9**	**86.4**	**85.9**

Table 7: The results of different strategies of sampling positions for replacing, where 'D-*' denotes the F score of detection and 'C-*' denotes the F score of correction.

Model	Replacing Strategy	Whole Set		Multi-typo Set	
		D-F	C-F	D-F	C-F
cBERT	-	85.3	89.5	84.8	83.9
cBERTNoise	Random	85.9	89.6	85.1	84.1
	ConfusionSet	**86.1**	**90.9**	**86.4**	**85.9**

Table 8: The results of different replacing strategies, where 'D-*' denotes the F score of detection and 'C-*' denotes the F score of correction.

Model	**Detection Score**			**Correction Score**		
	P	R	F	P	R	F
BERT	75.8	85.5	80.4	94.7	80.9	87.3
BERTCopy	**78.1**	**85.8**	**81.8**	**95.7**	**82.1**	**88.4**
cBERT	83.0	**87.8**	85.3	96.0	83.9	89.5
cBERTCopy	**84.0**	87.7	**85.6**	**96.8**	**84.8**	**90.4**

Table 9: The performances of incorporating the copy block on BERT and cBERT.

Model	Whole Set(%)		Multi-typo Set(%)	
	D-F	C-F	D-F	C-F
cBERT	85.3	89.5	84.8	83.9
MultiRound	85.0	90.2	86.1	85.1
NoseTrain	85.2	89.7	85.4	84.5
cBERTNoise	**86.1**	**90.9**	**86.4**	**85.9**

Table 10: Results of different methods for multi-typo texts on SIGHAN15.

during pre-training on large corpus. As a consequence, Bert-based models tend to overcorrect an infrequent but valid expression to a more frequent expression. To solve this problem, we propose to incorporate the copy mechanism in our correction model. Besides *cBERT*, we also investigate the effects of the copy block on *BERT* to further demonstrate its effectiveness. Table 9 illustrates the results. We observe that with the incorporation of copy block, both *BERTCopy* and *cBERTCopy* achieve better performances. Moreover, *cBERTCopy* achieves less improvements than *BERTCopy*. This phenomenon occurs because *cBERT*(Liu et al., 2021) is pre-trained for CSC, thus the overcorrection problem is not as serious as that in *BERT*.

4.9 Comparison of Different Methods for Multi-typo Texts

In this subsection, we implement another two methods for multi-typo texts. (1) *MultiRound* is based on cBERT but repeatedly corrects a given text in multiple rounds until no error could be detected. (2) *NoiseTrain* is also based on cBERT, but is trained with the noisy texts generated by the noisy block. Table 10 presents the results. We observe that all these methods could achieve improvements on the multi-typo dataset. However, *MultiRound* and *NoiseTrain* fail to achieve improvements on the whole set, which indicates that these methods have negative effects on single-typo or zero-typo texts. On contrast, *cBERTNoise* achieves significant improvements on both test set. This result demonstrates the effectiveness of the proposed framework for multi-typo texts.

In *NoiseTrain*, the constructed noise itself is involved in the correction loss during training, which changes the ratio of positive and negative samples in the training corpus. Moreover, the quality of the constructed noise also will affect the correction model significantly. However, in the proposed approach *cBERTNoise*, the constructed noise only serves as a context (see Equation 8). Therefore, the aforementioned two problems no longer exist. We

believe this is the most important factor that enables our model to achieve much better performance.

5 Conclusions

In this work, we first point out two limitations of previous CSC models, which are called *Contextual Typo Disturbance* problem and *Overcorrection* problem. To solve the first problem, we propose the noise modeling module to generate noisy context in training process. Experimental results show that this module is effective on multi-typo texts. To solve the *Overcorrection* problem, we incorporate a copy block in the correction model, which encourages our model to prefer to keep the input character when the miscorrected and input characters are both valid in the given context. Experimental results demonstrate its effectiveness on both BERT and cBERT. Moreover, the proposed model *CRASpell* outperforms all compared models and achieve new state-of-the-art performances on SIGHAN dataset.

References

Haithem Afli, Zhengwei Qiu, Andy Way, and Páraic Sheridan. 2016. Using SMT for OCR error correction of historical texts. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 962–966, Portorož, Slovenia. European Language Resources Association (ELRA).

Zuyi Bao, Chen Li, and Rui Wang. 2020. Chunk-based Chinese spelling check with global optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2031–2040, Online. Association for Computational Linguistics.

Jill Burstein and Martin Chodorow. 1999. Automated essay scoring for nonnative english speakers. In *Computer mediated language assessment and evaluation in natural language processing*.

Chao-Huang Chang. 1995. A new approach for automatic chinese spelling correction. In *Proceedings of Natural Language Processing Pacific Rim Symposium*, volume 95, pages 278–283. Citeseer.

Kuan-Yu Chen, Hung-Shin Lee, Chung-Han Lee, Hsin-Min Wang, and Hsin-Hsi Chen. 2013. A study of language modeling for Chinese spelling check. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 79–83, Nagoya, Japan. Asian Federation of Natural Language Processing.

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. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 871–881, Online. Association for Computational Linguistics.

Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016. Neural network translation models for grammatical error correction. *arXiv preprint arXiv:1606.00189*.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Rui Dong and David A Smith. 2018. Multi-input attention for unsupervised ocr correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2363–2372.

Rahhal Errattahi, Asmaa El Hannani, and Hassan Ouahmane. 2018. Automatic speech recognition errors detection and correction: A review. *Procedia Computer Science*, 128:32–37.

Jianfeng Gao, Chris Quirk, et al. 2010. A large scale ranker-based system for search query spelling correction. In *23rd International Conference on Computational Linguistics*.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065, Melbourne, Australia. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

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 2019)*, pages 160–169, Hong Kong, China. Association for Computational Linguistics.

Changning Huang, Haihua Pan, Zhou Ming, and Lei Zhang. 2000. Automatic detecting/correcting errors in chinese text by an approximate word-matching algorithm.

Li Huang, Junjie Li, Weiwei Jiang, Zhiyu Zhang, Minchuan Chen, Shaojun Wang, and Jing Xiao. 2021. PHMOSpell: Phonological and morphological knowledge guided Chinese spelling check. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5958–5967, Online. Association for Computational Linguistics.

Jianshu Ji, Qinlong Wang, Kristina Toutanova, Yongen Gong, Steven Truong, and Jianfeng Gao. 2017. A nested attention neural hybrid model for grammatical error correction. pages 753–762.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Piji Li and Shuming Shi. 2021. Tail-to-tail non-autoregressive sequence prediction for Chinese grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4973–4984, Online. Association for Computational Linguistics.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. *arXiv preprint arXiv:2106.14448*.

Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2019. A BERT-based universal model for both within- and cross-sentence clinical temporal relation extraction. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 65–71, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Chao-Lin Liu, Min-Hua Lai, Yi-Hsuan Chuang, and Chia-Ying Lee. 2010. Visually and phonologically similar characters in incorrect simplified Chinese words. In *Coling 2010: Posters*, pages 739–747, Beijing, China. Coling 2010 Organizing Committee.

Shulin Liu, Tao Yang, Tianchi Yue, Feng Zhang, and Di Wang. 2021. PLOME: Pre-training with misspelled knowledge for Chinese spelling correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2991–3000, Online. Association for Computational Linguistics.

Deryle Lonsdale and Diane Strong-Krause. 2003. Automated rating of esl essays. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing*, pages 61–67.

Bruno Martins and Mário J Silva. 2004. Spelling correction for search engine queries. In *International Conference on Natural Language Processing (in Spain)*, pages 372–383. Springer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. pages 3982–3992.

Arup Sarma and David D Palmer. 2004. Context-based speech recognition error detection and correction. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 85–88.

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*, pages 32–37, Beijing, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, 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*, pages 2517–2527, Brussels, Belgium. 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 Annual Meeting of the Association for Computational Linguistics*, pages 5780–5785, Florence, Italy. Association for Computational Linguistics.

Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013. Chinese spelling check evaluation at sighan bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. pages 72–77.

Junjie Yu and Zhenghua Li. 2014. Chinese spelling error detection and correction based on language model, pronunciation, and shape. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 220–223.

Liang-Chih Yu, Lung-Hao Lee, Yuen-Hsien Tseng, and Hsin-Hsi Chen. 2014. Overview of SIGHAN 2014 bake-off for Chinese spelling check. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 126–132, Wuhan, China. Association for Computational Linguistics.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 882–890, Online. Association for Computational Linguistics.