WSpeller: Robust Word Segmentation for Enhancing Chinese Spelling Check

Fangfang Li[1], **Youran Shan**[1], **Junwen Duan**[1,*], **Xingliang Mao**[2], **MinLie Huang**[3]
[1]School of Computer Science and Engineering, Central South University
[2]Institute of Big Data And Internet Innovation, Hunan University of Technology and Business
[3]Beijing National Research Center for Information Science and Technology, Tsinghua University
{lifangfang,shanyouran,jwduan}@csu.edu.cn
xingliangmao0929@163.com
aihuang@tsinghua.edu.cn

Abstract

Chinese spelling check (CSC) detects and corrects spelling errors in Chinese texts. Previous approaches have combined character-level phonetic and graphic information, ignoring the importance of segment-level information. According to our pilot study, spelling errors are always associated with incorrect word segmentation. When appropriate word boundaries are provided, CSC performance is greatly enhanced. Based on these findings, we present WSpeller, a CSC model that takes into account word segmentation. A fundamental component of WSpeller is a W-MLM, which is trained by predicting visually and phonetically similar words. Through modification of the embedding layer's input, word segmentation information can be incorporated. Additionally, a robust module is trained to assist the W-MLM-based correction module by predicting the correct word segmentations from sentences containing spelling errors. We evaluate WSpeller on the widely used benchmark datasets SIGHAN13, SIGHAN14, and SIGHAN15. Our model is superior to state-of-the-art baselines on SIGHAN13 and SIGHAN15 and maintains equal performance on SIGHAN14.

1 Introduction

Chinese Spelling Check (CSC) aims to detect and correct spelling errors in Chinese text, and can be used in many other natural language processing (NLP) applications, including search optimization (Martins and Silva, 2004; Gao et al., 2010), location extraction (Middleton et al., 2018), optical character recognition (OCR) (Afli et al., 2016), Automatic Speech Recognition (ASR) (Chao and Chang, 2020) and text classification (Xu et al., 2021).

Typos in the text are mostly phonetic or graphic of the correct character (Liu et al., 2010). Previous work focus on how to use the features of pronunciation and shape (Cheng et al., 2020; Ji et al., 2021). However, little attention has been devoted to the incorrect word segmentation induced by typos. As shown in Table 1, several typos not only affect semantic but also affect word segmentation. In this case, we perform a preliminary experiment detailed in Sec 3 to determine the impact of word segmentation on CSC. Experimental results indicate that if the appropriate word boundaries are provided, CSC performance will improve dramatically. However, it is also challenging to achieve robust word segmentations in the presence of typos.

State	Text
Correct	比起 前段 时间 已经 很 轻松 了
Wrong	比 七天段 时间 已经 很 轻 送 了
Translation	It's easier than it's been for a while

Table 1: An example of Chinese spelling error. Typos and the right characters are highlighted in red, while the spaces within the text show the word boundaries.

To cope with the problem, we propose WSpeller (**W**ord **S**peller), which consists of two submodules, namely the word segmentation module (WS-Module) and the correction module (C-Module). The WS-Module predicts the right word boundaries based on the original text (may have typos). Following that, the segment-level information is provided to C-Module for improving correction. The two submodules adopt different text encoding modules, and the hidden state of the WS-Module is passed to the C-Module, where the two submodules interact and are trained in a multi-task manner for mutual benefit. W-MLM (**W**ord-**M**asked **L**anguage **M**odel) is proposed as the text encoding module of C-Module because it adapts BERT (Devlin et al., 2019) with different embedding information and special replacement strategies and pre-trained on a large Chinese Corpus. WSpeller

*Corresponding Author

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 1179–1188
December 7-11, 2022

is evaluated against three widely used datasets: SIGHAN13 (Wu et al., 2013), SIGHAN14 (Yu et al., 2014), SIGHAN15 (Tseng et al., 2015). The experimental results show that WSpeller improves the performance of CSC. WSpeller increased the F1 scores by 3% and 2.1% in SIGHAN13 and SIGHAN15 and achieved almost the same performance as the state-of-the-art on SIGHAN14. Ablation studies reveal that both the word segmentation and the W-MLM contribute considerably to the performance. The contribution of this paper is summarized as follows:

- We are among the first to use the robust word segmentation in CSC, and our findings confirm its efficacy.
- We propose W-MLM to improve task adaptability, in which the embedding and replacement strategy of the masked language model is tailored to the specific needs of the CSC task.
- Experiments on the SIGHAN benchmark datasets demonstrate that our method outperformed strong competitors.

2 Related Work

In recent years, CSC has received widespread attention. Early works focus on rules and confusion sets to deal with CSC, follows pipelines, including detection, candidate generation and selection (Xie et al., 2015; Xin et al., 2014; Zhang et al., 2015; Chen et al., 2013). These methods are limited by manually set rules and fixed confusion sets.

With the development of deep learning, most methods focus on how to integrate the phonetic and graphic features of characters into the model. Hong et al. (Hong et al., 2019) proposed FASPell, which exploited the phonetic and graphic information to exclude candidates with low similarity. Cheng et al. (Cheng et al., 2020) proposed SpellGCN, which employed GCN to combine the phonetic and graphic information as the output layer. Ji et al. (Ji et al., 2021) proposed SpellBERT, which used GCN to model the Pinyin and radical of characters as the embedding layer. Wang et al. (Wang et al., 2021) proposed DCN, which added phonetic embedding when generating candidates, used attention mechanism to calculate the fraction between adjacent characters, and obtained the best path. Bao et al. (Bao et al., 2020) proposed to use semantic

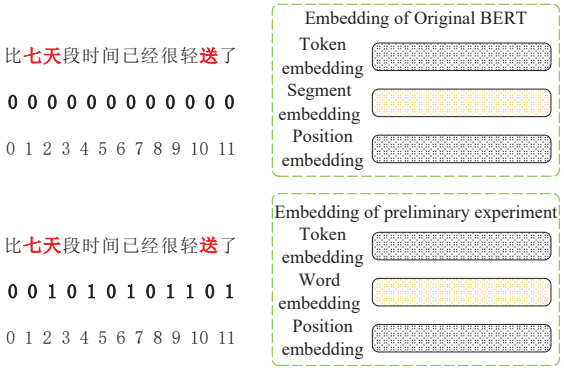

Figure 1: Change of the embedding layer. The sequence corresponding to word embedding represents the word segmentation of the text.

candidates to expand the confusion set and use a block-based structure to correct errors.

Some methods focus on training skills and the positive impact of detection on correction. Zhang et al. (Zhang et al., 2020) adopted GRU as detection network and BERT as correction network, and propose a soft masking strategy. Gan et al. (Gan et al., 2021) propose to judge the difficulty of learning samples through loss and apply self-supervised curriculum learning to CSC.

In recent years, the methods focus on integrating the similarity between characters into the model. However, they ignore the importance of word segmentation to CSC. In this paper, the word segmentation information of text is integrated into the model through pre-training and fine-tuning. Only a simple model structure can achieve a good correction effect.

3 Preliminary Experiment

Our primary premise is that the presence of typos hurts word segmentation and the ability correct would improve if better word boundaries were available. To validate this, we conduct a preliminary experiment in which we inject the precise word segmentation into BERT. As shown in Figure 1, the boundaries of words are represented by 0s and 1s, where 1 indicates segmentation is required before the current character. The word segmentation is obtained by LAC [1]. The result is shown in Table 2. We find that even a simple BERT can improve the F1 score of detection and correction by 6.2% and 7.4%, with precise word boundaries, implying that word boundaries play a significant role in the CSC.

[1]https://pypi.org/project/LAC/

Method	Detection Level			Correction Level		
	Pre	Rec	F1	Pre	Rec	F1
BERT	75.8	74.8	75.2	73.4	72.4	72.9
Seg	**79.9**	**79.9**	**79.9**	**78.3**	**78.3**	**78.3**

Table 2: Results of preliminary experiments. **BERT** means the conventional bert-based method. **Seg** means the method to change the input information of the embedding layer.

We, therefore, propose WSpeller, which includes word segmentation information for CSC. However, obtaining correct word boundaries for sentences containing typos is a challenge. WSpeller has been further enhanced with a robust WS-Module so that it can predict the word boundaries correctly.

4 Problem Definition

CSC aims to detect and correct typos in a given text $X = (x_1, x_2, \cdots, x_n)$, and finally get the detection sequence $D = (d_1, d_2, \cdots, d_n)$ representing whether corresponding x_i is a typo and the correction $Y = (y_1, y_2, \cdots, y_n)$ without spelling error.

5 The Proposed Model

In this section, we present the WSpeller, whose overall architecture is illustrated in Figure 2. WSpeller consists of a WS-Module (5.1) and a C-Module (5.3). While the WS-Module aims to predict the correct word boundaries in spite of spelling errors, the C-Module aims to correct the typos in conjunction with the WS-Module. To fully exploit the word segmentation information, the C-Module uses a pre-trained language model W-MLM (5.2).

5.1 Word Segmentation Module

In this section, WS-Module which aims to predict the word segmentations of corrected text from source text is introduced in detail. Given a source text X, we can easily get the hidden state $\boldsymbol{H}$ by BERT. Then we can get the predicted word segmentation result $S = (s_1, s_2, \cdots, s_n)$ by word segmentation network. The method is as follows:

$$\boldsymbol{H} = \text{BERT}(X) \quad (1)$$

$$S = \text{Softmax}(W_1 H + b_1) \quad (2)$$

where W_1 and b_1 are learnable matrices. Each item $s_i \in \{0, 1\}$ in S indicates the word boundaries, and $s_i = 1$ means that word segmentation is required before the current character x_i. In the training phase, we use LAC to segment the target text Y as the label of the WS-Module.

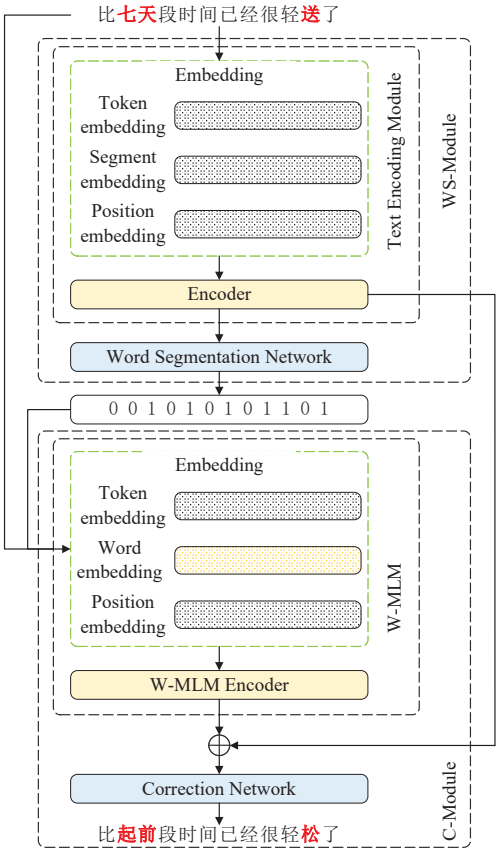

Figure 2: Overview of the WSpeller. WS-Module aims to predict the word segmentation of the target text based on the source text. After that, the predicted word segmentation result is used in C-Module, and finally obtain the target text.

5.2 Pre-training W-MLM

The text encoding module of the C-Module is W-MLM, whose overall structure is shown in Figure 3. The skeleton of W-MLM is BERT. Since the text is always single sentence input in CSC, the W-MLM ignores the next sentence prediction (NSP).

The key points of W-MLM lie in three parts, including replacement strategy selection, replacement character generation and word segmentation information integration. Replacement strategy selection and replacement character generation are used to generate training sets.

5.2.1 Replacement Strategy Selection

In order for W-MLM to learn the knowledge of continuous errors, misuse of correct words and similarity between characters, different from the conventional replacement , we introduce the re-

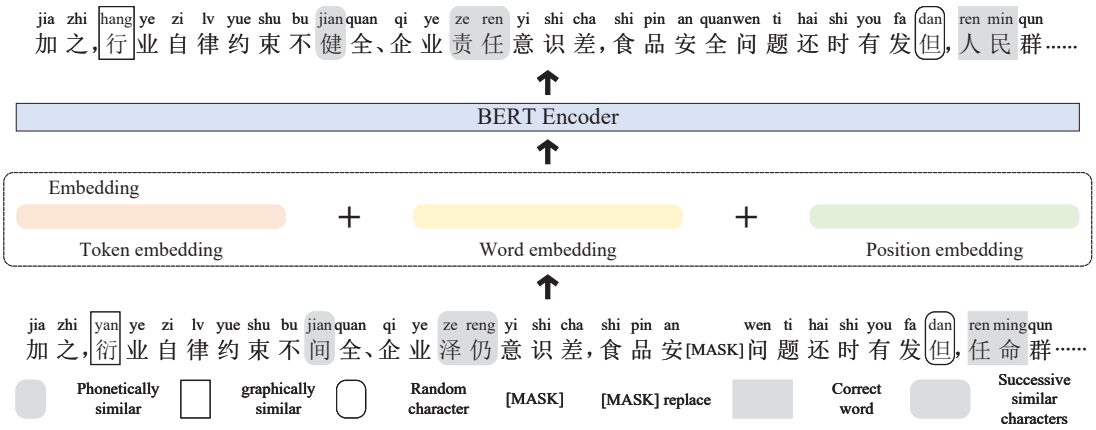

Figure 3: Overview of W-MLM.

Strategy	Prob	Example
Origin&Same	10%	跟我一起(qi)去爬山吗
[MASK]	20%	跟我一[MASK]去爬山吗
Random	10%	跟我一里(li)去爬山吗
Graphics	10%	跟我一超(chao)去爬山吗
Pronunciation	30%	跟我一气(qi)去爬山吗
Continuous	10%	跟我仪器(yi qi)去爬山吗
	10%	跟我已棋(yi qi)去爬山吗

Table 3: Probability of each replacement strategy being selected. The replaced characters are marked in red, and the pronunciation of it is in parentheses. **Origin&Same** means the original text and the same character replacement. **[MASK]** means [MASK] replacement. **Random** means random character replacement.**Graphics** means graphically similar character replacement. **Pronunciation** means phonetically similar character replacement. **Continuous** means continuous similar character replacement and correct word replacement.

placement of continuous similar characters, correct words and similar characters.

In accordance with BERT, we randomly select 15% of the characters in the text for replacement. Based on empirical evidence, we establish the probability of a certain replacement strategy being used for a particular character. Table 3 shows the probability of each strategy being chosen. As shown in Figure 3, W-MLM is finally trained to predict its original text based on the partially replaced text.

5.2.2 Replacement Character Generation

We are prone to mistype characters as similar and commonly used characters when using input methods. Inspired by this, we comprehensively consider similarity and commonality to generate candidates.

For a character ch to be replaced, its initial candidate list is the vocabulary. Firstly, we obtain the phonetic and graphic similarity between each candidate and ch by calculating the edit distance of ideographic description sequence (IDS) and Pinyin sequence (Hong et al., 2019). At the same time, the frequency of candidates is counted in SogouCA[2].

Then, we initialize the phonetic candidate list $Candidate_p = (cp_1, cp_2, \cdots, cp_k)$ and graphic candidate list $Candidate_g$ for ch according to the rule of preferentially taking the characters with higher similarity and higher frequency. We empirically set k to 30, and update the frequency list $Count^p = (count_1^p, count_2^p, \cdots, count_k^p)$ and $Count^g$. After that, we comprehensively consider similarity and commonality to obtain the phonetic score $ScoreP = (sp_1, sp_2, \cdots, sp_m)$ and graphic score $ScoreG$. $ScoreG$ is calculated in the same way as $ScoreP$. Take sp_i as an example:

$$score_i^c = \frac{count_i^p - min(Count^p)}{max(Count^p) - min(Count^p)} \quad (3)$$

$$score_i^s = Pronunciation(cp_i, ch) \quad (4)$$

$$sp_i = w_c \times score_i^s + w_s \times score_i^s \quad (5)$$

where $max(Count^p)$ and $min(Count^p)$ is the Max and Min in $Count^p$. $score_i^c$, $score_i^s$ is the the frequency and similarity score. Pronunciation is to calculate the pronunciation similarity between cp_i and ch. w_c and w_s is the weights of $score_i^c$, sim_i^p. We empirically set them to 0.3 and 0.7.

Finally, the higher the score sp_i of cp_i, the more likely cp_i is to be used to replace ch in phonetically similar character replacement, as are other replacement strategies.

[2]http://www.sogou.com/labs/resource/ca.php

5.2.3 Word Segmentation Information Integration

In WSpeller, the text is input as a single sentence. Therefore, the segment sequence of the original BERT for the sentence number is actually all zero. Obviously, it does not contain any information useful for correction. In this regard, we integrate word segmentation information into W-MLM by replacing segmentation embedding with word embedding. The word segmentation results S of the text before random replacement is obtained by LAC. Thus, the model can learn the changes of embedding layer in the pre-training stage, so that the model can perform better in the fine-tuning stage.

5.3 Correction Module

As shown in Figure 2, the C-Module is composed of W-MLM and correction network. Different from the WS-Module, we have obtained the predicted word segmentation result S at this stage. We assume that S is the word segmentation of the target text Y, and this information will play a great role in correction.

Hidden state $\tilde{\boldsymbol{H}}$ can be obtained by token sequence T, word segmentation S, and position sequence P. In addition, we believe that C-Module should have a positive impact on WS-Module. Therefore, we add the H of the WS-Module and the $\tilde{H}$ of the C-Module as the final hidden state $\tilde{\boldsymbol{H}}'$. After that, the target text Y is further obtained:

$$\tilde{\boldsymbol{H}} = W\text{-}MLM(T, S, P) \quad (6)$$

$$\tilde{\boldsymbol{H}}' = H + \tilde{H} \quad (7)$$

$$\tilde{\boldsymbol{H}}'' = LayerNorm(GELU(W_2\tilde{H}' + b_2)) \quad (8)$$

$$Y = Softmax(W_3\tilde{H}'' + b_3) \quad (9)$$

where W_2, W_3, b_2, b_3 are learnable parameter, LayerNorm and GELU represent the layer normalization and activation functions in BERT, respectively.

5.4 Learning

WS-Module and C-Module are jointly trained. The learning process is driven by optimizing correction and word segmentation respectively.

$$\mathcal{L}_s = -\sum_{i=1}^{n} logP_s(s_i = truth_{si}|X) \quad (10)$$

$$\mathcal{L}_c = -\sum_{i=1}^{n} logP_c(y_i = truth_{yi}|X) \quad (11)$$

$$\mathcal{L} = \lambda \times \mathcal{L}_s + (1 - \lambda) \times \mathcal{L}_c \quad (12)$$

Training Data	Sentences	ASL	Typos
Hybrid	271,329	42.6	381,962
SIGHAN13	700	41.8	343
SIGHAN14	3,437	49.6	5,122
SIGHAN15	2,338	31.3	3,037
Total	277,804	42.6	390,464
Testing Data	**Sentences**	**ASL**	**Typos**
SIGHAN13	1,000	74.3	1,224
SIGHAN14	1,062	50.0	771
SIGHAN15	1,100	30.6	703
Total	3,162	50.9	2,698

Table 4: Statistics of the datasets. ASL is the average length of sentences

where $\mathcal{L}_s$ is the loss of WS-Module, $truth_{si}$ is the correct result of s_i. $\mathcal{L}_c$ is the loss of C-Module, $truth_{yi}$ is the correct result of y_i. $\lambda \in [0, 1]$ is the weight of $\mathcal{L}_s$, which represents the focus of model training. When λ is closer to 1, it means that more attention is paid to the WS-Module.

6 Experiments

6.1 Dataset and Metrics

In the pre-training stage, we used many pre-processed data from Wikipedia used in TaCL (Su et al., 2021). In the fine-training, the data used for training includes all the data in Hybrid (Wang et al., 2018) and the training sets in SIGHAN13 (Wu et al., 2013), SIGHAN14 (Yu et al., 2014), and SIGHAN15 (Tseng et al., 2015). To evaluate, we use the test sets of SIGHAN13, SIGHAN14, and SIGHAN15. Since the SIGHAN is traditional, we use the processed data[3] which follow the previous work (Cheng et al., 2020) and convert them to Simplified Chinese using the OpenCC tool[4]. The statistics for the datasets are shown in Table 4.

We report Precision, Recall, and F1 scores at sentence level in detection and correction, which are commonly used in the CSC. Sentence level means that the current sentence can only be judged to be correct if all typos have been detected and corrected.

6.2 Baseline

We compare WSpeller with five typical baselines.

[3]https://github.com/DaDaMrX/ReaLiSe
[4]https://github.com/BYVoid/OpenCC

Dataset	Method	Detection Level			Correction Level		
		Pre	Rec	F1	Pre	Rec	F1
SIGHAN13	FASPell (Hong et al., 2019)	76.2	63.2	69.1	73.1	60.5	66.2
	SpellGCN (Cheng et al., 2020)	80.1	74.4	77.2	78.3	72.7	75.4
	SpellGCN* (Wang et al., 2021)	85.2	77.7	81.2	83.4	76.1	79.6
	DCN* (Wang et al., 2021)	**86.8**	79.6	83.0	**86.7**	77.7	81.0
	BERT*	81.7	86.0	83.8	79.7	83.9	81.8
	WSpeller*	82.3	**86.9**	**84.6**	81.2	**85.7**	**83.4**
SIGHAN14	FASPell (Hong et al., 2019)	61.0	53.5	57.0	59.4	52.0	55.4
	SpellGCN (Cheng et al., 2020)	65.1	69.5	67.2	63.1	67.2	65.3
	DCN (Wang et al., 2021)	67.4	**70.4**	**68.9**	65.8	**68.7**	**67.2**
	BERT	66.9	63.9	65.4	64.6	61.7	63.1
	WSpeller	**70.4**	66.3	68.3	**69.0**	65.0	67.0
SIGHAN15	FASPell (Hong et al., 2019)	67.6	60.0	63.5	66.6	59.1	62.6
	Soft-Masked BERT (Zhang et al., 2020)	73.7	73.2	73.5	66.7	66.2	66.4
	SpellGCN (Cheng et al., 2020)	74.8	80.7	77.7	72.1	77.7	75.9
	DCN (Wang et al., 2021)	77.1	**80.9**	79.0	74.5	**78.2**	76.3
	BERT	78.7	74.5	76.5	75.8	71.7	73.7
	WSpeller	**81.9**	78.0	**79.9**	**79.9**	76.1	**77.9**

Table 5: Results of Precision, Recall, and F1 scores(%). * indicates that "的", "得", "地" are ignored when calculating results on SIGHAN13, and the results of DCN* and SpellGCN* are reported by DCN (Wang et al., 2021).

- FASPell (Hong et al., 2019) measures the similarity between characters and filters characters with low similarity.
- SpellGCN (Cheng et al., 2020) uses GCN to model the similarity between characters and integrates the knowledge into the output layer.
- Soft-Masked BERT (Zhang et al., 2020) proposes a Soft-Mask layer that adjusts the embedding according to the probability of typos.
- SpellBERT (Ji et al., 2021) uses GCN to model the Pinyin and radical of characters as the embedding layer.
- DCN (Wang et al., 2021) adds pronunciation information when generating candidates, and models the correlation between adjacent characters to select the best candidates.

6.3 Experiment Setting

We use one Tesla V100 for pre-training and two GeForce RTX 2080Ti for follow-up experiments. In the fine-tuning stage, maximum sequence length is set to 160, batch size is set to 22, learning rate is set to 5e-5, training epoch is set to 20, warmup is set to 0.1, λ is set to 0.2, optimization method is Adam. In pre-training stage, batch size is set to 32, learning rate is set to 1e-4, training epoch is set to 1, optimization method is Adam. In addition, there are lots of labeling errors about "的", "得", "地" in SIGHAN13, which affect the evaluation, so we follow the previous work (Wang et al., 2021) to ignore all corrections about "的", "得", "地".

Furthermore, since the ability of W-Module is poor at the beginning of training, it will affect the training of C-Module. Therefore, we use scheduled sampling to provide a high proportion of correct word segmentation in the initial phase of training, and then slowly decrease the proportion until it reaches zero. In the first epoch, the probability of providing the correct word segmentation results is set to 0.9 and decreases to 0 in the 10th epoch.

6.4 Main Results

Table 5 illustrates the detection and correction performance on SIGHAN13, SIGHAN14, and SIGHAN15 of the proposed method and baseline models. As shown in Table 5, WSpeller significantly outperforms the other methods and the results show the effectiveness of our method and the enhancement of word segmentation for CSC.

Soft-Masked BERT uses a Soft-Masked layer

Method	Detection Level				Correction Level			
	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
SpellGCN (Cheng et al., 2020)	83.7	85.9	80.6	83.1	82.2	85.4	77.6	81.3
SpellBERT (Ji et al., 2021)	-	87.5	73.6	80.0	-	87.1	71.5	78.5
DCN (Wang et al., 2021)	84.6	**88.0**	80.2	83.9	83.2	**87.6**	77.3	82.1
BERT	82.4	85.1	77.8	81.3	80.9	84.6	74.9	79.4
WSpeller	**84.7**	87.1	**81.0**	**83.9**	**83.6**	86.8	**78.7**	**82.6**

Table 6: Accuracy, Precision, Recall and F1 scores(%) at sentence level in detection and correction evaluated by SIGHAN15 official tools.

to smooth the hidden state with typo probability. FASPell, SpellGCN, and DCN use different methods to integrate the similarity between characters to achieve better results. Different from these methods, WSpeller focuses on the occurrence of typos, which often leads to word segmentation errors. Compared with DCN, the F1 score of correction improves by 2.4% and 1.6% on SIGHAN13 and SIGHAN15. Meanwhile, competitive results are obtained on SIGHAN14. The further case study is given in Sec 6.6. At the same time, compared with BERT, the F1 scores improve significantly on SIGHAN13, SIGHAN14, and SIGHAN15, suggesting that the integration of word segmentation information has improved WSpeller over the basic model.

Further, we follow the previous works (Cheng et al., 2020; Wang et al., 2021) and use the official tool[5] to evaluate the performance of WSpeller on SIGHAN15. The results are shown in Table 6. SpellBERT uses GCN to model the features of characters as the embedding layer. Compared with it, the F1 scores of WSpeller in detection and correction are increased by 3.9% and 4.1%. At the same time, WSpeller achieves the best result among all methods, which further indicates the effectiveness of WSpeller.

6.5 Ablation Study

Ablation study is presented to understand how the components of WSpeller and the hyperparameter influence the performance. The metrics reported in this section are averaged over SIGHAN13, SIGHAN14, and SIGHAN15.

6.5.1 Effect of Each Module

We remove the following components from WSpeller to study their contributions to the overall performance, and the results are shown in Table 7.

Method	Detection Level		
	Pre	Rec	F1
BERT	75.8	74.8	75.2
WSpeller	**78.2**	**77.1**	**77.6**
-W-MLM	76.4	75.9	76.1
-Different encoding	76.8	75.6	76.1
-Schedule sampling	77.2	76.5	76.8
LAC	77.9	77.1	77.4
Method	**Correction Level**		
	Pre	Rec	F1
BERT	73.4	72.4	72.9
WSpeller	**76.7**	**75.6**	**76.1**
-W-MLM	74.5	74.1	74.2
-Different encoding	75.1	74.0	74.5
-Schedule sampling	76.2	75.5	75.8
LAC	75.9	75.1	75.5

Table 7: Ablation experiment results of Precision, Recall and F1 scores (%). **BERT** is BERT-based method. **WSpeller** is our proposed method. **-W-MLM** is to replace the W-MLM module with BERT. **-Different encoding** is to use the same text encoding module in WS-Module and C-Module. **-Schedule sampling** removes the training strategy. **LAC** integrates word segmentation information from the source text.

As shown in Table 7, W-MLM contributes the most to WSpeller's correction ability. If W-MLM is removed, the F1 score of detection and correction will decrease by 1.5% and 1.9%, respectively. This suggests that learning the similarity between characters in the pre-training stage and adapting to changes in the embedding layer information has a positive effect on CSC.

LAC introduces the word segmentation information of the source text on the basis of BERT. Since

[5]http://nlp.ee.ncu.edu.tw/resource/csc.html

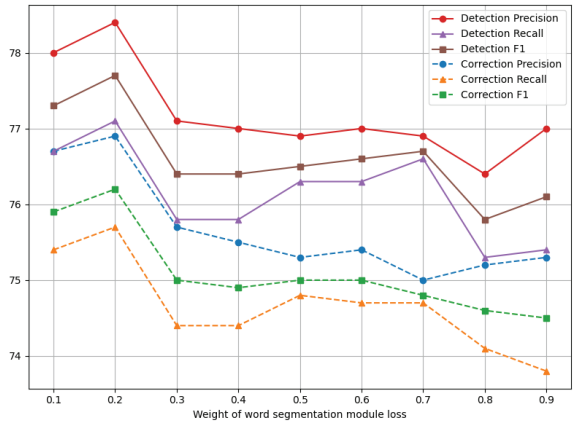

Figure 4: Experimental results on hyperparameter. The sentence level Precision, Recall, and F1 scores (%) are reported on the detection and correction. The abscissa is the value of λ, that is, the weight of WS-Module loss. Solid lines represent the detection subtask, dashed lines represent the correction subtask.

No.	mean	sentences
1	source	应 为 他 学得 很好...
	target	因为 他 学得 很好...
	predict	因为 他 学得 很好...
	translate	Because he learned it well...
2	source	...我 要 跟旁 东 说 一声...
	target	...我 要 跟旁 东 说 一声...
	predict	...我 要 跟 房东 说 一声...
	translate	...I have to tell the landlord...
3	source	...有 一个 刻板 观念...
	target	...有 一个 刻版 观念...
	predict	...有 一个 刻板 观念...
	translate	...There is a stereotype...

Table 8: Example of WSpeller's corrected results. Parts of text are omitted by ellipsis because the text is too long. Typos and their correct characters are marked in red and green. Spaces in the text represent the word boundaries. The word boundaries of source and target come from LAC, and the word boundaries of predict come from WSpeller.

the integrated word segmentation information contains some errors, its correction ability is worse than that of WSpeller, but higher than that of BERT. This indicates that the integration of word segmentation information is effective, and the accuracy of word segmentation information will affect the final correction effect, and WSpeller has stronger word segmentation ability in texts with typos.

6.5.2 Effect of Hyperparameter

In the joint training, $\lambda \in [0, 1]$ represents the weight of WS-Module loss. When λ is large, the model will pay more attention to the training of WS-Module. Empirically, word segmentation is simpler than correction, so the weight of the WS-Module should be less than that of the C-Module in joint training.

We set λ to nine different values. When λ is set to a different value, the effect of WSpeller is shown in Figure 4. As shown in Figure 4, the Precision, Recall, and F1 score of the correction and detection of WSpeller reached the best when $\lambda = 0.2$. On the whole, when the loss of WS-Module accounts for small weight, WSpeller has better performance.

6.6 Case Study

Three representative examples of WSpeller are shown in Table 8. In the first example, the typo led to the wrong word segmentation. WSpeller not only correct the typo, but also get the correct word boundaries. In the second example, the typo led to the wrong word segmentation, but it is not labeled in the dataset. WSpeller correct the typo and get the correct word segmentation result. In the third example, "板" is correct, but is incorrectly labeled as "版". In the last two examples, the prediction results of WSpeller are correct. However, due to mislabeling, these correct prediction would not improve the F1 scores, but reduce it. For further analysis, we manually count the proportion of text containing label errors according to the badcases of WSpeller.

In SIGHAN13, SIGHAN14, and SIGHAN15, 28.2%, 35.3%, and 30.6% of badcases contain label errors, which greatly affects the evaluation. The proportion of label errors in SIGHAN14 is the highest, which may be one of the reasons why the effect of WSpeller in SIGHAN14 is not satisfactory. Several examples show that WSpeller has better word segmentation ability when typos exist. Meanwhile, integrating word segmentation information contributes to the correction ability.

Further, according to our statistics, of the typos in three datasets, about 52.5%, 42.2%, and 39.5%, respectively, resulted in word segmentation errors. WSpeller can correct 84.3%, 66.6%, and 63.6% error word boundaries only based on source text respectively. This indicates that a considerable part of typos will affect word segmentation, and WSpeller has good word segmentation ability even

if there are typos in the text.

7 Conclusion

In this paper, we have proposed WSpeller, a novel end-to-end framework for CSC. Unlike most of the previous methods, which included the phonetic and graphic information of characters in the model through various methods, we emphasize the importance of robust word segmentation in the CSC task. We added an additional WS-Module to predict the correct word boundaries for sentences with typos. The experimental results also suggest that adding word segmentation information to the model can improve the performance.

In the future, we will investigate other methods for providing word segmentation information to the model for correction. We will also explore how to use WSpeller for Chinese semantic corrections.

8 Limitations

In our study, we discovered that correct word segmentation would improve the ability of correction, but achieving accurate word segmentation from source texts that contain typos remains a challenge. However, although WSpeller is capable of segmenting words with high accuracy, there is still a lot of room for improvement. By having more accurate word boundaries available, the model's correction capabilities can be further enhanced, as the upper bound in the preliminary experimental results in Section 3. As WSpeller encodes text twice, its inference speed is limited. A future attempt will be made to improve WS-Module to reduce its complexity.

Acknowledgements

This research is supported by National Natural Science Foundation of China [62172449, 62006251], Hunan Provincial Natural Science Foundation of China [2021JJ30870,2022JJ30211], Changsha Municipal Natural Science Foundation [kq2202300]. And this research was carried out in part using computing resources at the High Performance Computing Center of Central South University.

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 2016, Portorož, Slovenia, May 23-28, 2016*. 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, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 2031–2040. Association for Computational Linguistics.

Yu-Chieh Chao and Chia-Hui Chang. 2020. Automatic spelling correction for ASR corpus in traditional chinese language using seq2seq models. In *International Computer Symposium, ICS 2020, Tainan, Taiwan, December 17-19, 2020*, pages 553–558. IEEE.

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, SIGHAN@IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*, pages 79–83. 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, ACL 2020, Online, July 5-10, 2020*, pages 871–881. Association for Computational 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.

Zifa Gan, Hongfei Xu, and Hongying Zan. 2021. Self-supervised curriculum learning for spelling error correction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3487–3494. Association for Computational Linguistics.

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.

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.

Tuo Ji, Hang Yan, and Xipeng Qiu. 2021. Spellbert: A lightweight pretrained model for chinese spelling check. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3544–3551. 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, 23rd International Conference on Computational Linguistics, Posters Volume, 23-27 August 2010, Beijing, China*, pages 739–747. Chinese Information Processing Society of China.

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.

Stuart E. Middleton, Giorgos Kordopatis-Zilos, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2018. Location extraction from social media: Geoparsing, location disambiguation, and geotagging. *ACM Trans. Inf. Syst.*, 36(4).

Yixuan Su, Fangyu Liu, Zaiqiao Meng, Lei Shu, Ehsan Shareghi, and Nigel Collier. 2021. Tacl: Improving BERT pre-training with token-aware contrastive learning. *CoRR*, abs/2111.04198.

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.

Baoxin Wang, Wanxiang Che, Dayong Wu, Shijin Wang, Guoping Hu, and Ting Liu. 2021. Dynamic connected networks for chinese spelling check. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2437–2446. Association for Computational Linguistics.

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.

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, SIGHAN@IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*, pages 35–42. Asian Federation of Natural Language Processing.

Weijian Xie, Peijie Huang, Xinrui Zhang, Kaiduo Hong, Qiang Huang, Bingzhou Chen, and Lei Huang. 2015. Chinese spelling check system based on n-gram model. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing, SIGHAN@IJCNLP 2015, Beijing, China, July 30-31, 2015*, pages 128–136. Association for Computational Linguistics.

Yang Xin, Hai Zhao, Yuzhu Wang, and Zhongye Jia. 2014. An improved graph model for chinese spell checking. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing, Wuhan, China, October 20-21, 2014*, pages 157–166. Association for Computational Linguistics.

JunLi Xu, JiaHui Hao, XiMo Bian, and XiaoMei Wang. 2021. Multi-task fine-tuning on bert using spelling errors correction for chinese text classification robustness. In *2021 IEEE 4th International Conference on Big Data and Artificial Intelligence (BDAI)*, pages 110–114.

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, Wuhan, China, October 20-21, 2014*, pages 126–132. Association for Computational Linguistics.

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, 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.