Chunk-based Chinese Spelling Check with Global Optimization

Zuyi Bao, Chen Li and **Rui Wang**
Alibaba Group
{zuyi.bzy,puji.lc,masi.wr}@alibaba-inc.com

Abstract

Chinese spelling check is a challenging task due to the characteristics of the Chinese language, such as the large character set, no word boundary, and short word length. On the one hand, most of the previous works only consider corrections with similar character pronunciation or shape, failing to correct visually and phonologically irrelevant typos. On the other hand, pipeline-style architectures are widely adopted to deal with different types of spelling errors in individual modules, which is difficult to optimize. In order to handle these issues, in this work, 1) we extend the traditional confusion sets with semantical candidates to cover different types of errors; 2) we propose a chunk-based framework to correct single-character and multi-character word errors uniformly; and 3) we adopt a global optimization strategy to enable a sentence-level correction selection. The experimental results show that the proposed approach achieves a new state-of-the-art performance on three benchmark datasets, as well as an optical character recognition dataset.

1 Introduction

Spelling check is a task to automatically detect and correct spelling errors in human writings. Spelling check is well-studied for languages such as English, and many resources and tools have been developed. However, the characteristics of the Chinese language make the Chinese spelling check (CSC)[1] quite different from the English one in three aspects:

- In contrast to English words that are composed of a small set of Latin letters, Chinese has more than three thousand frequently used characters. The large character set leads to a huge search space for the CSC models.
- For English spelling check, the basic unit is the word. However, Chinese characters are continuously written without word delimiter, and the word definition varies across different linguistic theories (Xue, 2003; Emerson, 2005). It makes the sentence with spelling errors more ambiguous, and more challenging for the spell checkers to detect and correct the errors.
- Chinese words usually consist of one to four characters and are much shorter than the English word. Spelling errors can drastically change the meaning of the word. Thus, the CSC task relies on the contextual semantic information to find the best correction.

For the first challenge, previous research demonstrates that most of the Chinese spelling errors come from similar pronunciations, shapes, or meanings (Liu et al., 2011; Chen et al., 2011). Previous CSC models usually employ the characters with similar pronunciation or shape as the confusion set to reduce the search space, but the visually and phonologically irrelevant typos cannot be handled. Recent work aims at replacing the pronunciation and shape confusion sets with a dynamically generated confusion set by masked language models, which retrieve the semantically related candidates according to the contextual information (Hong et al., 2019). However, due to the lack of knowledge about human errors, masked language models correct the spelling errors ignoring the pronunciation or shape similarity. Therefore, combining the two comes as a natural solution.

For the second challenge, early works rely on the segmentation results from a Chinese word segmentation system (Yu and Li, 2014). However, as the

[1] As Chinese spelling check involves both error detection and correction, we do not distinguish between spelling check and spelling correction in this paper.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 2031–2040
November 16 - 20, 2020.

segmentation system is trained on the clean corpus, the spelling errors often lead to incorrect segmentation results. The accumulated errors make the spell checking even more difficult. Thus, character-based models are proposed to perform the correction at the character-level directly, which are more robust to segmentation errors (Zhang et al., 2015; Hong et al., 2019; Zhang et al., 2020). However, the character-based model cannot effectively utilize the word-level semantic information, and the correction is also more difficult to interpret. In order to explore and utilize the word-level information, the word-based methods are designed to do word segmentation and spelling error corrections jointly. Previous works show that the word-based correction models often perform better than their character-based counterparts (Jia et al., 2013; Hsieh et al., 2015; Yeh et al., 2015; Zhao et al., 2017). Since word-based correction models usually apply a pipeline of submodules and handle special cases (e.g., single-character words) individually, the complex architecture makes it difficult to perform global optimization.

For the third challenge, previous works mainly rely on the local context features such as point-wise mutual information (PMI), part-of-speech (POS) n-gram, and perplexity from an n-gram language model (Liu et al., 2013; Zhang et al., 2015; Yeh et al., 2015). As these statistical features are limited within a fixed-size window, it is difficult to capture the deep contextual information.

In the paper, we propose a unified framework combining features and benefits from previous works. We employ confusion sets from similar pronunciations, shapes, and semantics to deal with different types of spelling errors. A chunk-based decoding approach is proposed to model both single-character and multi-character words in a uniform way. We also finetune an error model based on the large-scale pretrained language model to include deep semantic information. A global optimization algorithm is adopted to combine different features and select the best correction. The experiment results show that the proposed approach achieves a new state-of-the-art performance on the three benchmark datasets. A further experiment shows that our method is also effective for optical character recognition (OCR) errors. Our contributions are summarized as follows:

1. We propose a chunk-based decoding method with global optimization to correct single-character and multi-character word typos in a unified framework.
2. We combine pronunciation, shape, and semantic confusion sets to handle different spelling errors.
3. Our method achieves new state-of-the-art performance on the three benchmark datasets and an OCR dataset.

2 Approach

The workflow of the proposed approach is shown in Figure 1. The proposed spelling check method adopts the chunk-based decoding, which processes single-character and multi-character words in a uniform way. During decoding, the candidates with variable length are dynamically generated according to the input sentence and the partially decoded sentence. For selecting the best correction, a global ranking optimization is used to combine different features.[2]

2.1 Chunk-based Decoding

The chunk-based decoding treats single-character words, multi-character words, phrases, and idioms equivalently as chunks. It provides a unified framework where we can easily extend the candidate generation methods. The framework also makes the implementation of global optimization to be possible. Given an input sentence with n characters $s = [c_1, c_2, \cdots, c_n]$, the chunk-based decoding gradually segments and corrects the input sentence at the same time. It attempts to find the best combination of chunk candidates and rewrites the input sentence to its correction in a left-to-right style:

$$s_c = \underset{\hat{s} \in L(s)}{\arg\max} f(\hat{s}, s) \tag{1}$$

where f is a scoring function. s is the input sentence, and $L(s)$ refers to the set of all possible combinations of chunk candidates for s.

The decoding process employs the framework of the beam search algorithm (Lowerre, 1976), and the details are shown in Algorithm 1. The beam is initialized with an empty correction. In the loop, we extend each partially decoded correction in the beam with dynamically generated chunk candidates. A scoring model is utilized for giving each

[2] The CSC task only considers substitution errors as spelling errors and leaves other errors to grammatical errors (Hsieh et al., 2015).

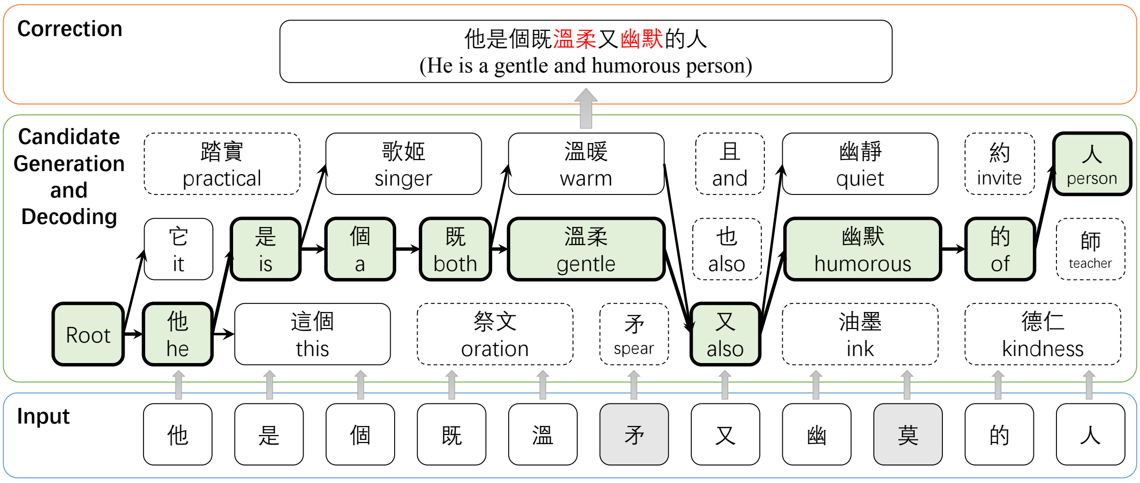


Figure 1: The workflow of the proposed chunk-based decoding method during the inference time. The chunk-based candidate generation and decoding are used to disambiguate and correct the input sentence gradually.

Algorithm 1: Chunk-based Decoding

```
Input: Input sentence $s$, Beam size $k$, Vocabulary $V$
Output: The corrected sentence $s_c$
Init beam ← [Root];
Init temp ← [];
Init cands ← None;
Init x ← None;
while Any correction in beam is not finished do
    temp ← [];
    foreach correction in beam do
        if correction is finished then
            temp.append(correction);
            continue;
        end
        cands ← get_candidates($s$, correction, $V$);
        foreach candidate in cands do
            x ← correction.extend(candidate);
            x.score ← score(x);
            temp.append(x);
        end
    end
    sort_prune_beam(temp, $k$);
    beam ← temp;
end
$s_c$ ← beam[0];
Return $s_c$
```

correction a confidence score. The details about the candidate generation and correction selection will be introduced in Section 2.2 and 2.3. At the end of each loop, we sort the beam and prune the corrections with low confidence to reduce the search space. Finally, after every correction in the beam decodes the whole input sentence, we output the most confident correction as the final result.

Essentially, the decoding stage jointly searches all possible segmentations and their corrections. From another point of view, the decoding gradually disambiguates and rewrites the sentence.

2.2 Candidate Generation

Previous work proposes to retrieve the candidates according to pronunciation or shape confusion sets (Liu et al., 2011; Chen et al., 2011). Following these works, we adopt confusion sets to reduce the search space. For handling single-character word typos and visually or phonologically irrelevant typos, we extend the pronunciation and shape confusion sets with semantic confusion set.

The candidate generation module assumes that each span of characters in the input sentence can be misspelled. According to confusion sets from three aspects, we generate all possible chunk candidates for the partially decoded correction. Given a vocabulary V, an input sentence s, and a start position i, we consider chunks of characters starting at i and within a max length as a potential typo and generate possible correction candidates:

Pronunciation: Given a chunk of characters $chunk_{ij} = [c_i, \cdots, c_j]$ from the i-th to the j-th character in the sentence s, we convert $chunk_{ij}$ to its pinyin[3] and retrieve all the candidates in a similar pronunciation from the V.

Shape: In addition to pronunciation, we also consider the candidates in a similar shape. Within a $chunk_{ij}$, we substitute characters with their visually similar characters and keep the candidates that can be found in the V. In practice, making a balance between speed and quality, we only consider candidates that have 1 edit distance (1 substitution) with the $chunk_{ij}$.

Semantic: Beyond the pronunciation and shape

[3] Pinyin is the official phonetic system for transcribing the sound of Chinese characters into Latin script.

similarity, we also utilize language models to retrieve semantically reasonable candidates according to the contextual information. Specifically, we employ the masked language model (Devlin et al., 2018) as it is effective for modeling long-range dependencies. Following Hong et al. (2019), we finetune the pretrained masked language model on the CSC training data and use the top k prediction of each character as the semantic confusion set. For candidates generation, we substitute each character in the $chunk_{ij}$ with its semantically similar characters and keep the candidates that can be found in the V. Similar to shape confusion set, in practice, we only consider candidates that have 1 edit distance (1 substitution) with the $chunk_{ij}$.

2.3 Correction Selection

In this section, we introduce the training strategy for correction selection and the features we used for global optimization. Most of the previous work follows the noisy channel model (Brill and Moore, 2000), which formulates the error correction tasks as:

$$s_c = \arg\max_{\hat{s}} p(\hat{s}|s) \tag{2}$$

where the s is the input sentence, and $\hat{s}$ refers to a possible correction. The formula can be further rewritten through the Bayes rule as:

$$s_c = \arg\max_{\hat{s}} \frac{p(s|\hat{s}) \cdot p(\hat{s})}{p(s)} \tag{3}$$

where $p(s|\hat{s})$ and $p(\hat{s})$ refer to the error model probability and the sentence probability respectively. Then we omit the $p(s)$ as it is constant for every $\hat{s}$ and take logarithm:

$$s_c = \arg\max_{\hat{s}} (\log p(s|\hat{s}) + \log p(\hat{s})) \tag{4}$$

The formula becomes a linear model combining the error model probability and the sentence probability in logarithm. In practice, the error model and the sentence probability is complex. In the experiment, we use a bundle of features and apply a linear model as the score function for approximation.

$$score = \sum_i w_i \cdot \text{feat}_i \tag{5}$$

where w_i is the weight for i-th feature feat$_i$.

Name	Description
ed	the character-level edit distance between s and $\hat{s}$.
pyed	the edit distance between the pinyin of s and $\hat{s}$.
n-chunk	the number of chunks in $\hat{s}$.
wlm	the perplexity of $\hat{s}$ measured by a word-level n-gram language model.
cem	the improvement of log probability from a character error model.
n-py	the number of chunks that are from the pronunciation confusion set.
n-shape	the number of chunks that are from the shape confusion set.
n-lm	the number of chunks that are from the semantic confusion set.

Table 1: The features used for the correction selection. s and $\hat{s}$ refer to the input sentence and a correction.

The features we used for correction selection are listed with their descriptions in Table 1. The *ed* and *pyed* are used to calculate the similarity of the correction and input sentence through character-level and pronunciation-level. A longer chunk is usually more unambiguous than a shorter one, thus a correction with less *n-chunk* is often more reasonable. The *wlm* is used for checking the fluency of a correction. The *n-py*, *n-shape* and *n-lm* assign weights to different confusion sets. The *cem* is used for modeling the character-level error probability. We directly use the finetuned masked language model in the semantic confusion set as the error model. When a chunk of characters $[c_i, \cdots, c_j]$ is substituted with $[\hat{c}_i, \cdots, \hat{c}_j]$, we calculate the chunk-level *cem* approximately as:

$$cem = \sum_{k=i}^{j} (\log p(\hat{c}_k|c_k, s) - \log p(c_k|c_k, s)) \tag{6}$$

where $p(\hat{c}_k|c_k, s)$ is the probability of replacing c_k with $\hat{c}_k$ given the input sentence s.[4]

For combining different features, we apply the Minimum Error Rate Training (MERT) algothrim (Och, 2003). Given the top n outputs, the MERT algorithm optimizes the scoring function by learning to rerank the decoded sentences according to their similarity to the gold sentence. Rather than a local ranking, the MERT algorithm measures the similarity directly by sentence-level metrics to achieve a global optimization.

3 Experiments

In the following sections, we will introduce the datasets and the experimental settings first, and

[4] Note that we use $p(\hat{c}_k|c_k)$ to simulate the error model $p(s|\hat{s})$, because our error model is contextualized and the calculation costs will be huge if we calculate $p(s|\hat{s})$ for each candidate.

Dataset	Training Data			Test Data		
	# Sent.	Error Rate	Avg. Length	# Sent.	Error Rate	Avg. Length
csc_{13}	700	50.0%	41.81	1000	99.6%	74.3
csc_{14}	3437	99.9%	49.6	1062	49.8%	50.0
csc_{15}	2339	100.0%	31.3	1100	50.0%	30.6
ocr	3575	100.0%	10.1	1000	100%	10.2

Table 2: Statistics of datesets. The error rate refers to the percentage of sentences with errors.

then the performance on the three benchmark datasets is listed to show the effectiveness of the proposed method. Finally, the evaluation of an OCR subtitle dataset shows that our method can be adapted to OCR errors as well.

3.1 Setup

We evaluate the proposed method on three CSC benchmark datasets and an OCR subtitle error correction dataset. The three CSC datasets are from SIGHAN13 (Wu et al., 2013), CLP14 (Yu et al., 2014) and SIGHAN15 (Tseng et al., 2015), and the OCR dataset is released from Hong et al. (2019). For simplicity, we denote the CSC datasets from SIGHAN13, CLP14, SIGHAN15 and OCR subtitles as csc_{13}, csc_{14}, csc_{15} and ocr, respectively. The csc_{13} and ocr dataset is evaluated on edit-level with the official evaluation tool from SIGHAN13. Following the official setting, the csc_{13} dataset adopts different test set for error dectection and correction. The csc_{14} and csc_{15} dataset are evaluated on sentence-level with the official evaluation tool from CLP14 and SIGHAN15 respectively.[5] Following previous work, we combine the training data from csc_{13}, csc_{14} and csc_{15} as our training set for csc dataset. The training set of ocr dataset is used to learn the model for the OCR dataset. The statistics of the datasets are listed in Figure 2. The ocr dataset contains only erroneous sentences and has a significantly shorter sentence length comparing to the csc datasets.

For the candidate generation phase, the vocabulary V used in the experiments is collected from gigaword corpus (LDC2011T13) and Chinese idioms. For csc dataset, we segmented the traditional Chinese corpus in the gigaword with hannlp[6] and keep the words that appear more than 10 times in the corpus. For ocr dataset, we use the simplified Chinese part for generating vocabulary V. For the pronunciation confusion set, we use pypinyin[7] for conversion between Chinese characters and pinyin. For the shape confusion set, we use the released one from SIGHAN13. For the semantic confusion set, we finetune the released Chinese version of the mask language model BERT (Devlin et al., 2018) on the CSC training set with the officially released Tensorflow code.[8] We also experimented with the whole word masking variants, such as BERT-wwm (Cui et al., 2019), but it did not show a significant improvement. The batch size, learning rate, and training epoch of the finetuning are set to 32, $2e^{-5}$, and 3, respectively. We use the top 5 output as the semantic candidates. The max length of chunks is set to 6 to cover most of the cases. For chunks with one character, we only keep the semantic candidates to reduce the false alarm rate.

For the correction selection phase, the beam size used in the experiment is set to 10. The segmented gigaword corpus is also used for training a traditional Chinese and a simplified Chinese n-gram word language model through kenlm.[9] For the MERT algorithm, we initialize the weights of the score function with zero and use the implement from Z-MERT (Zaidan, 2009). For optimization, we output the top 10 results and set the maximum MERT iterations to 15. The bilingual evaluation understudy (BLEU) is used as the training metric as it calculates the sentence-level similarity and often leads to better precision.

3.2 Experiment Results on the CSC Datasets

We first report the performance of the proposed method on the csc_{13}, csc_{14} and csc_{15} dataset. As shown in Table 3, when comparing to previous strong CSC systems, our proposed chunk-based method achieves a significant improvement on the three datasets.

Zhao et al. (2017) employ a graph-based model and integrate spelling checking with word segmentation. However, their proposed method only pro-

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

[6] https://github.com/hankcs/HanLP

[7] https://github.com/mozillazg/python-pinyin

[8] https://github.com/google-research/bert

[9] https://github.com/kpu/kenlm

Dataset	Model	Detection Level				Correction Level			
		Acc	P	R	F1	Acc	P	R	F1
csc_{13}	Yeh et al. (2015)	74.80	44.31	37.67	40.72	66.30	70.30	62.50	66.17
	Zhao et al. (2017)	-	-	-	-	37.00	70.50	35.60	47.31
	Hong et al. (2019)*	-	-	-	-	60.5	73.1	60.5	66.2
	Cheng et al. (2020)‡	-	55.90	46.99	51.06	-	44.58	37.47	40.72
	our method	**83.20**	**61.19**	**75.67**	**67.66**	**67.20**	**74.34**	**67.20**	**70.59**
csc_{14}	Zhao et al. (2017)	-	-	-	-	-	55.50	39.14	45.90
	Hong et al. (2019)	**70.0**	61.0	53.5	57.0	**69.3**	59.4	**52.0**	55.4
	Cheng et al. (2020)‡	-	58.27	54.53	56.28	-	51.01	47.65	49.27
	our method	**70.0**	**78.65**	**54.80**	**64.59**	68.08	**77.43**	51.04	**61.52**
csc_{15}	Zhang et al. (2015)	70.09	80.27	53.27	64.04	69.18	79.72	51.45	62.54
	Hong et al. (2019)	74.2	67.6	60.0	63.5	73.7	66.6	59.1	62.6
	Zhang et al. (2020)	**80.9**	73.7	**73.2**	**73.5**	**77.4**	66.7	**66.2**	66.4
	Cheng et al. (2020)‡	-	70.97	64.00	67.30	-	60.08	54.18	56.98
	our method	76.82	**88.11**	62.00	72.79	74.64	**87.33**	57.64	**69.44**

Table 3: The main results on csc_{13}, csc_{14} and csc_{15} datasets. *The csc_{13} detection-level performance of Hong et al. (2019) is obtained on the test set of correction task and thus incomparable with the results from other work. The results with ‡ are reproduced by rerunning the released code and evaluation scripts on the standard CSC datasets. The Wang et al. (2018) and Wang et al. (2019) calculate the performance on the character-level, which makes their results incomparable with other works.

cesses the multi-character words. Two types of single-character words are handled by rules and an individual module. The separated modules make their system difficult to fully explore the annotated data and obtain a global optimization.

Zhang et al. (2015) combine the character-level candidate generation with a two-stage filter model. For the first stage, they use a logistic regression classifier to reduce the size of candidates. In the second stage, they utilize the online translation system and search engine to select the best correction. Although they get help from empirically developed online systems for correction selection, our approach outperforms them, indicating the effectiveness of the chunk-based framework.

Hong et al. (2019) finetune the pretrained BERT as a character-based correction model and filter the visually/phonologically irrelevant corrections to improve precision. In other words, they employ a character-level candidate generation and perform a locally optimized character-based correction selection. In the experiment, our method outperforms Hong et al. (2019) with a large margin, which indicates the effectiveness of the globally optimized chunk-based decoding.

Zhang et al. (2020) propose to train a detection and a correction network jointly. In the experiment, although they employ 5 million pseudo data for extra pretraining, the proposed method still obtains an improved performance on the correction level.

Cheng et al. (2020) propose to incorporate phonological and visual confusion sets into the CSC models through a graph convolutional network. As the performance reported in their paper is obtrained with external training data, we reproduced their results on the standard CSC datasets by rerunning their released code and evaluation scripts.

3.3 Experiment Results for the OCR Errors

We also evaluate our approach on the OCR subtitle error correction dataset, and the results are listed in Table 4. For the error detection level, the proposed method achieves a significant improvement over the previous model from Hong et al. (2019). The *ocr* dataset has a shorter average sentence length. The finetuned BERT model does not have enough context to obtain semantically accurate corrections. Hong et al. (2019) only generate the candidates according to the BERT model and obtain a low recall. The proposed method is more robust to short sentences because we also employ the confusion sets from pronunciation and shape.

For the correction-level, we also observe a significant improvement in the F1 score. However, we notice that our method obtains a lower precision comparing with Hong et al. (2019). We analyzed and found that the OCR subtitles are extracted from

Model	Detection Level				Correction Level			
	Acc	P	R	F1	Acc	P	R	F1
Hong et al. (2019)	18.6	**78.5**	18.6	30.1	17.4	**73.4**	17.4	28.1
our method	**63.30**	77.57	**63.30**	**69.71**	**37.90**	46.45	**37.90**	**41.74**

Table 4: The results on the OCR subtitle error correction dataset *ocr*.

Model	Correction Level		
	P	R	F1
all	87.33	**57.64**	**69.44**
all - pinyin	87.54	54.91	67.49
all - shape	86.81	57.45	69.15
all - semantic	**88.33**	48.18	62.35

Table 5: The results on csc_{15} dataset of disabling different confusion sets. The *pinyin*, *shape*, *semantic* refers to the pronunciation, shape, semantic confusion set, respectively.

the entertainment domain, which contains many named entities and is quite different from the news vocabulary we used. Thus, although we detected the spelling errors, it is difficult to retrieve the correct candidate. We leave the domain adaptation problem to future work.

3.4 Analysis of Confusion Sets

To reveal the contributions of each confusion set, we conduct experiments to disable each confusion set one at a time. The experiment results are listed in Table 5. The results show that, without the pronunciation confusion set, the proposed method suffers a obvious drop on the recall rate. The shape confusion set only brings a slight improvement, which is explained that errors in similar shape only count for a small part of the spelling errors in human writings. Another significant improvement comes from the semantic confusion set. With a small sacrifice in precision, we observe an obvious increment of recall rate. This experiment result shows that the semantic confusion set is a good complement to the traditional candidate generation.

3.5 MERT v.s. BERT

In this section, we compare the locally optimized character-based correction model with our globally optimized chunk-based approach. In the experiment, we use the finetuned BERT checker (Hong et al., 2019) as the character-based model. We use the test set of csc_{15} and compare the performance on the recall rate of the single-character errors and multi-character errors individually. The single-character error refers to the misspelling of a single character, and we take a chunk of continual typos containing more than one character as the multi-character error. On the test set, the recall rate is calculated at the chunk-level, and the experiment results are shown in Figure 2. The recall of the BERT checker model almost comes from single-character errors. For the multi-character errors, the proposed method obtains a significantly better performance, which indicates the effectiveness of globally optimized chunk-based decoding.

In Figure 3, we list two cases and their corrections from the BERT checker and our method. The BERT checker takes the CSC task as a character

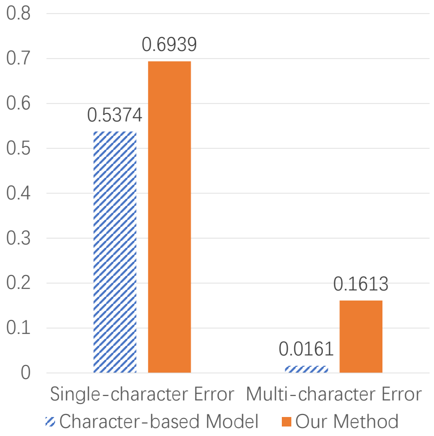
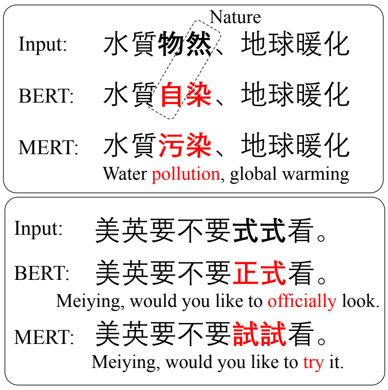


Figure 2: The comparison of recall between a locally optimized character-based BERT checker and the proposed globally optimized chunk-based method.



Input: 水質物然、地球暖化

BERT: 水質自染、地球暖化

MERT: 水質污染、地球暖化

Water pollution, global warming

Input: 美英要不要式式看。

BERT: 美英要不要正式看。

Meiying, would you like to officially look.

MERT: 美英要不要試試看。

Meiying, would you like to try it.

Figure 3: The case analysis between the BERT checker and the proposed globally optimized method.

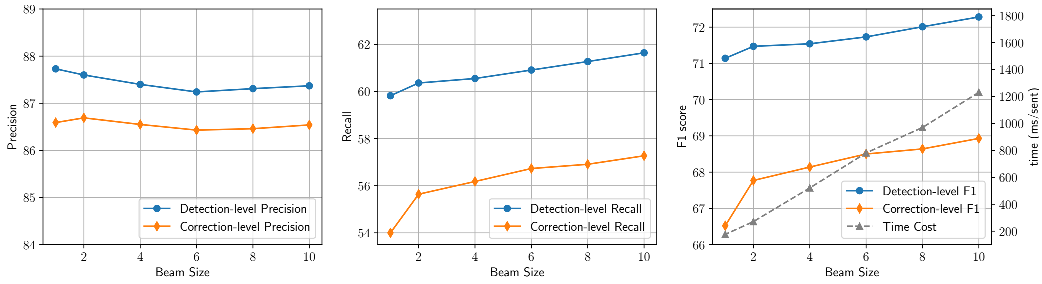


Figure 4: The precision, recall, F1 score and runtime on the csc_{15} dataset with different beam size.

Beam	Recall on Sentences with			
Size	1 Errors	2 Errors	3 Errors	3+ Errors
1	69.56	42.63	43.86	**26.83**
2	72.13 (+2.57)	44.21 (+1.58)	**45.61 (+1.75)**	26.83 (+0.00)
4	73.07 (+0.94)	**44.74 (+0.53)**	43.86 (-1.75)	26.83 (+0.00)
8	**73.30 (+0.23)**	44.74 (+0.00)	43.86 (+0.00)	26.83 (+0.00)

Table 6: The edit-level recall for sentences in the csc_{15} dataset with different beam size.

sequence labeling problem and adopts a character-wise local optimization (Hong et al., 2019). For the multi-character error, the BERT checker tends to correct the misspelled characters according to their incorrect context. As shown in the first case, the BERT checker correct 物 to 自 because 自 and the incorrect neighbour 然 can compose a word 自然 (nature). Thus, the BERT checker usually corrects only a part of the multi-character typo or rewrites the typo to a word which is unfitted in the sentence. The proposed method directly generates the candidates for a chunk of misspelled characters and performs a global optimization to replace the whole typo.

3.6 Beam Size

The proposed chunk-based decoding is constructed under the framework of beam search. In each loop step, the beam search algorithm prunes the size of candidates to a pre-defined beam size to reduce the search complexity.

In this section, we investigate how beam size influences the performance of the proposed CSC model. We run experiments with a range of beam size on the test set of csc_{15}, and the results and runtime are shown in Figure 4. When the beam size increases, the CSC model is able to obverse more candidates and obtains a significant improvement in the recall rate. At the same time, a larger search space brings more noise, which leads to a slight drop in precision. As a result, the F1 score achieves an improvement when the beam size increases. For the runtime, Figure 4 illustrates that the time-cost grows linearly against the beam size.

To further investigate the improvement of recall rate, we divide the test set according to the number of errors in the sentences and calculate the edit-level recall for the model under different beam sizes. As shown in Table 6, the experiment results illustrate that the main improvement of the recall rate comes from the sentences with only one error. As the larger beam size essentially includes a longer context, the experiment results demonstrate that CSC errors require more contextual information even for single-character errors. For sentences with more errors, the recall rate increases rapidly when the beam size is small (e.g., beam size from 1 to 2). However, the recall rate does not increase significantly after the beam grows to an appropriate size (e.g., a beam size of 4). This experiment result illustrates that, for sentences with multiple errors, the bottleneck comes from the candidate selection.

4 Related Work

Previous work of CSC is closely related to a series of shared tasks (Wu et al., 2013; Yu et al., 2014; Tseng et al., 2015). The workflow of CSC systems can be roughly divided into two phases, candidate generation and candidate selection.

For the candidate generation phase, most of the previous work retrieves the candidates according to pronunciation or shape (Liu et al., 2011; Chen et al., 2011; Yu and Li, 2014; Yeh et al., 2015). Recently, Hong et al. (2019) propose to replace the traditional confusion sets with a dynamically generated one. They treat the CSC as a sequence labeling problem and finetune a pretrained masked language model to generate candidates. For reducing the false alarm rate, they filter the result with pronunciation and shape similarity. Their method inspired us to finetune the masked language model for generating semantically related candidates.

For the candidate selection phase, the perplexity from language models is frequently used for selecting the most reasonable candidate (Chang, 1995; Liu et al., 2013; Jia et al., 2013; Yu and Li, 2014; Yeh et al., 2015). Rules are effective and often included in the CSC model for handling single-character errors (Hsieh et al., 2015; Zhang et al., 2015; Zhao et al., 2017). Recent researchers rely on supervised methods to achieve further improvement. The supervised error model is frequently involved in previous work (Hsieh et al., 2015; Yeh et al., 2015; Zhang et al., 2015). Liu et al. (2013) uses the support vector machines (SVMs) to rerank the candidate list. Yeh et al. (2015) employ a maximum entropy (ME) model for correction selection. Zhao et al. (2017) use conditional random fields (CRFs) to handle two types of misspelled single-character word. Cheng et al. (2020) propose to incorporate phonological and visual similarity knowledge into the CSC models via a graph convolutional network.

Due to the limited size of CSC training data, the supervised models suffer from the lack of annotated data. Liu et al. (2013) generate pseudo data by replacing the character in the training sentence with characters in the confusion set. Similarly, Zhang et al. (2020) generate homophonous pseudo data to pretrain the detection and correction network jointly. Web texts are in large quantities and contain more errors than published articles. Hsieh et al. (2015) propose to extract spelling error samples from the Google web 1T corpus. Wang et al. (2018) propose the OCR-based and ASR-based methods to mimic human errors. They further proposed a pointer network to model the CSC task under the framework of a seq2seq model (Wang et al., 2019).

5 Conlusion

In this work, we present a new framework for Chinese spelling check. We include the masked language model for generating semantically related candidates. The chunk-based decoding is employed to handle single-character and multi-character errors in a uniform way. A global optimization strategy is adopted for combining different features. The effectiveness of the proposed method is verified on three CSC benchmark datasets and an OCR subtitle dataset. As for the future work, we plan to extend the proposed framework to Chinese grammatical error correction and explore the possibilities of training in an end-to-end style.

References

Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong. 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*, volume 95, pages 278–283. Citeseer.

Yong-Zhi Chen, Shih-Hung Wu, Ping-Che Yang, Tsun Ku, and Gwo-Dong Chen. 2011. Improve the detection of improperly used chinese characters in students' essays with error model. *International Journal of Continuing Engineering Education and Life Long Learning*, 21(1):103–116.

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 *ACL*.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.

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.

Yu-Ming Hsieh, Ming-Hong Bai, Shu-Ling Huang, and Keh-Jiann Chen. 2015. Correcting chinese spelling errors with word lattice decoding. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 14(4):18.

Zhongye Jia, Peilu Wang, and Hai Zhao. 2013. Graph model for Chinese spell checking. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 88–92, Nagoya, Japan. Asian Federation of Natural Language Processing.

C-L Liu, M-H Lai, K-W Tien, Y-H Chuang, S-H Wu, and C-Y Lee. 2011. Visually and phonologically similar characters in incorrect chinese words: Analyses, identification, and applications. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(2):10.

Xiaodong Liu, Kevin Cheng, Yanyan Luo, Kevin Duh, and Yuji Matsumoto. 2013. A hybrid Chinese spelling correction using language model and statistical machine translation with reranking. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 54–58, Nagoya, Japan. Asian Federation of Natural Language Processing.

Bruce T Lowerre. 1976. The harpy speech recognition system. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

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.

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, Nagoya, Japan. Asian Federation of Natural Language Processing.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, pages 29–48.

Jui-Feng Yeh, Wen-Yi Chen, and Mao-Chuan Su. 2015. Chinese spelling checker based on an inverted index list with a rescoring mechanism. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 14(4):17.

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, Wuhan, China. Association for Computational Linguistics.

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.

Omar F. Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked bert. *arXiv preprint arXiv:2005.07421*.

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*, pages 38–45, Beijing, China. Association for Computational Linguistics.

Hai Zhao, Deng Cai, Yang Xin, Yuzhu Wang, and Zhongye Jia. 2017. A hybrid model for chinese spelling check. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(3):21.