Sparsity Regularization for Chinese Spelling Check

Anonymous ACL submission

Abstract

The Chinese Spelling Check (CSC) research objective is to detect and correct the spelling errors in the input. Generally, the number of incorrect characters in the input is far less than the correct, so the error probability sequence of input sentence predicted by the detection module should be sparse and sharp. However, all existing work has ignored this problem. In this paper, we add a sparsity regularization item to the objective function to make the output of the detection module close to sparse and sharp. We study two kinds of regularization: L_1 regularization and minimum entropy regularization. Extensive experiments on the SIGHAN show that the sparsity regularization proposed in this paper can effectively improve the performance of the CSC model while without increasing the computational complexity. In addition, the robustness experiment results show that our method is robust.

1 Introduction

Chinese Spelling Check (CSC) task is an important task in Natural Language Processing (NLP) community, which aims to correct spelling errors in Chinese input (Yu & Li, 2014). There are usually two types of errors in Chinese text editing and text Recognition: errors of visually similar characters caused by Optical Character Recognition (OCR) or Wubi input method, and errors caused by the misuse of phonologically similar characters (Duan Jianyong, 2021). Spelling errors will affect the semantics of the sentence and then negatively impact downstream text processing tasks. Therefore, it is necessary and practical to study the CSC.

The CSC task consists of two subtasks: error detection and error correction. For the CSC task, complete Chinese sentences with/without spelling errors will be given as the input. For the detection subtask, the detection module should return the locations of the incorrect characters. For the correction subtask, the correct module should point out the correct characters based on the output of detection subtask. The error correction problem is a follow-up problem of error detection for checking spelling errors (Hládek et al., 2020; Tseng et al., 2015).

The input of the detection module is Chinese sentence $X = (x_1, x_2, \cdots, x_n)$, and x_i represents the i-th character in the input. The output is a probability vector $P = (p_1, p_2, \cdots, p_n)$, p_i represents the probability that x_i is an incorrect character. In the input of the CSC, the number of incorrect characters is usually far less than correct characters. Therefore, the probability vector output by the detection module should be sparse and sharp. For example, as shown in Figure 1, for the input “我以前想要高诉你，可是忘了，真户秃。” (Correct sentence: “我以前想要告诉你，可是忘了，真糊涂。”, meaning is “I wanted to tell you before, but I forgot, so confused. ”), the best output of detection module is a probability vector which with 1 at the location of “高” (告, tell), and “户秃” (糊涂, confused), and 0 at the other positions. The best output of the detection module is sparse and sharp, but this is a tremendous challenge for the detection module.

Based on the above problem, and inspired by the attention with sparsity regularization (Zhang et al., 2018), this paper uses sparsity regularization item to constrain the probability vector output of the detection module. We study two kinds of sparsity regularization: L_1 regularization and minimum entropy regularization. L_1 regularization will constrain the sum of the absolute value of the probability vector output by

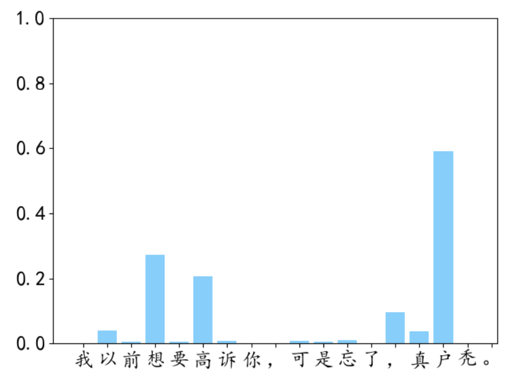

Figure 1: The example of the output of spelling detection.

detection module, so as to promote the sparsity of the probability vector. Minimum entropy regularization minimizes the entropy of the probability vector output by the detection module. As we all know, the more certain, the smaller entropy. So, when p_i is 0 or 1, the entropy of the probability vector P is the smallest. Therefore, minimum entropy regularization can make the probability vector output by the detection module close to sparse and sharp.

In order to test the effectiveness of our approach, we evaluated our method on the SIGHAN dataset. On the premise of using the same training set, validation set, and test set and keeping the same experimental setup, the two kinds of sparsity regularization used in this paper can effectively improve the performance of the CSC model. In addition, we use five new test sets to test the robustness of our approach. In these test sets, the proportion of without spelling errors sentences increases sequentially. The main contributions of this paper are as follows:

- We firstly propose use sparsity regularization to improve the performance of CSC model while without increasing the computational complexity of the model, and analyze why the sparsity regularization can be effective in CSC task from theoretical level.

- We study two kinds of sparsity regularization: L_1 regularization and minimum entropy regularization, and test the effectiveness and robustness of these two kinds of sparsity regularization through experiments.

2 Related Work

Early CSC methods were based on rules such as chunk, syntax, and grammar to determine spelling errors in the input (Hirst & Budanitsky, 2005).This kind of approach has high correction accuracy but poor generalization performance. Because the rules need to be defined by experts based on knowledge and experience, the rules can only cover limited situations, and the formulation of these rules takes a lot of time. Then the researchers tried to use the statistical language model to complete the CSC task. First, replace the characters in the input sentence sequentially according to the confusion set, then use the statistical language model to score these replaced sentences, and give correction suggestions based on the score (Huang et al., 2014; Yeh et al., 2017). But the method based on the statistical language model has shortcomings, that is, the expressive and learning capability of the model are poor.

Deep learning technology can contribute to this problem. The literature (Duan et al., 2019) employs Bi-LSTM combined with conditional random field (CRF) to achieve the error detection, and CRF was used to predict the best annotation sequence. The literature (Wang et al., 2021) employs Lattice-LSTM to dynamically integrate characters, words, and pinyin information, and then the CRF layer detected errors according to the integrated information. The focus of these works is to integrate multiple features to improve the performance of the model. Different from these works, the literature (Wang et al., 2019) applies the pointer-generation model, which excellent performance in automatic summarization tasks, to the error correction task. For each character in the input sentence, the pointer-generation model decides whether to copy the character or replace by the character generated by the Generation network according to the probability output by the detection module. As far as we know, this is the first CSC model using the soft strategy. However, these methods are difficult to adopt when the corpus is limited, because these models need enough labeled corpus for training (Tan et al., 2020).

In recent years, BERT has been applied to many natural language processing tasks as a pre-trained representation model and has shown strong performance (Devlin et al., 2019). Some researchers applied the BERT to the CSC task and achieved state-of-the-art (SOTA) results. The literature (Cheng et al., 2020) employs BERT to obtain the distributed representation of the input and employs the graph convolutional neural network (GCN) to learn the similarity knowledge in phonological and visual pronunciation to complete CSC task. The literature (Zhang et al., 2020) employs the Bi-GRU network as the detection module and then employs BERT to complete the error correction task according to the output of the detection module. The literature (Li et al., 2021) proposes a cloze-style detector-corrector framework (DCSpell) that firstly detects whether a character is erroneous before correcting it. DCSpell employs the discriminator of ELECTRA (Clark et al., 2016) as the Detector to detect the positions of incorrect characters and then employs BERT to correcting it.

The model based on BERT has achieved the SOTA on the CSC task, but some researchers pointed out that the error detection capability of BERT is poor (Zhang et al., 2020; Hong et al., 2019). Therefore, how to improve the detection capability of the CSC model base on BERT is a critical issue, which is also the focus of this paper.

3 Method

3.1 Soft-Masked BERT

The sparsity regularization method proposed in this paper is an improvement method that can improve the performance of the CSC model. The sparsity regularization method can be widely used in CSC tasks at a theoretical level. In order to show the superiority of our method, we use the strong baseline model Soft-Masked BERT (SM) (Zhang et al., 2020). The model structure is shown in Figure 2.

The SM model is composed of a detection module and correction module. The detection module uses a bidirectional GRU (Bi-GRU) model and the correction module uses BERT. The input of the detection module is the word embedding sequence $E = (e_1, e_2, \cdots e_n)$, e_i represents the embedding representation of the character x_i. The e_i consists of word embedding, position embedding, and segment embedding like the input of BERT. The output of the detection module is a probability vector $P = (p_1, p_2, \cdots p_n)$, p_i represents the probability that the character x_i is incorrect.

In the SM model, for each character of the sequence, the p_i is calculated as

$$p_i = P_d(g_i = 1|X) = \sigma\left(W_d h_i^d + b_d\right) \qquad (1)$$

where $P_d(g_i = 1|X)$ denotes the conditional probability predicted by the detection module, σ denotes the sigmoid function, h_i^d is the hidden state of the Bi-GRU, W_d and b_d are parameters of the model.

According to the probability output by the detection module, the soft-masked embedding e_i' for the x_i is calculated as **Formula** (2).

$$e_i' = p_i \cdot e_{\text{mask}} + (1 - p_i) \cdot e_i \qquad (2)$$

where e_i is the embedding of x_i outputted by BERT and e_{mask} is the mask embedding of character [MASK] outputted by BERT. When the p_i close to 0, the e_i' is close to the e_i; otherwise, it is close to the e_{mask}, which can reduce the impact of incorrect characters on the semantics of the input sentence.

The correction network is a sequential multi-class labeling task, which is based on the BERT model. The input of the correction network is the sequence $E' = (e_1', e_2', \cdots, e_n')$ that consists of soft-masked embedding. The output is a character sequence corrected by correction network, we define it as $Y = (y_1, y_2, \cdots y_n)$.

BERT is based on the transformer model and the hidden states of the output of BERT are denoted as $H^c = (h_1^c, h_2^c, \cdots, h_n^c)$. For each x_i, the corresponding output of the SM model is calculated as follows

$$P_c(\, y_i = j \mid X\,) = \text{softmax}(W h_i' + b)\,[j] \qquad (3)$$

Where $P_c(\, y_i = j \mid X\,)$ define the probability that x_i is corrected as character j in the output of the correct module, softmax denote the softmax function, W and b are parameters, h_i' is the hidden state which is obtained by residual connection, defined as

$$h_i' = h_i^c + e_i \qquad (4)$$

The training of the SM model is conducted end-to-end. For the dataset $D = \left\{\left(X^j, Y^j\right)\right\}_{j=1}^N$, X^j is the j-th input sequence, Y^j is the correct sequence of the X^j, N denotes that there are N with\without error sentences in the dataset. The goal of the training process is to optimize the two objective functions, corresponding to the detection module and error module.

$$\mathcal{L}_d = \sum_{j=1}^N \sum_{i=1}^n \log P_d\left(\, g_i^j \mid X_j, \theta_d\,\right) \qquad (5)$$

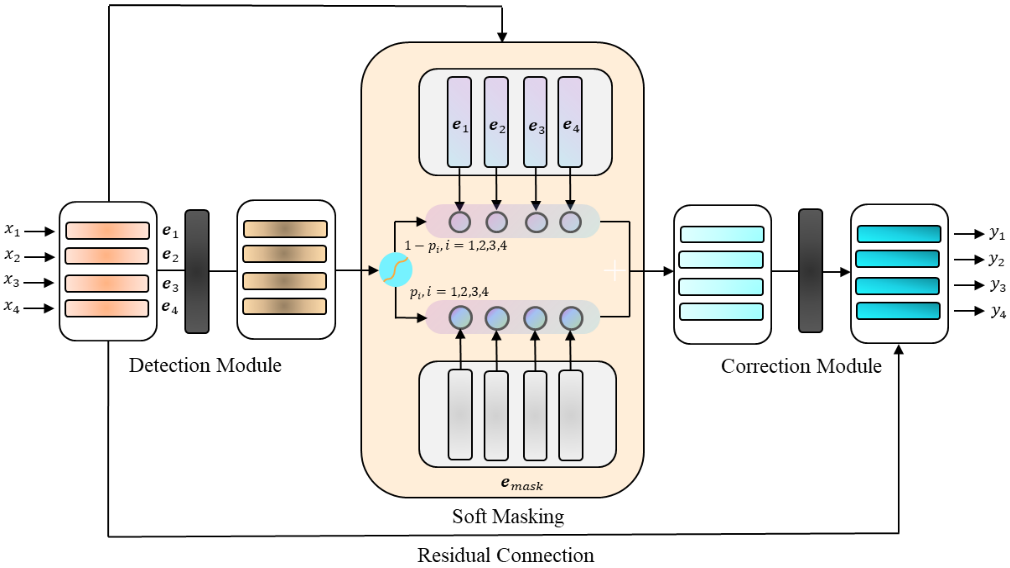

Figure 2: Structure of Soft-Masked BERT.

$$\mathcal{L}_c = \sum_{j=1}^{N} \sum_{i=1}^{n} \log P_c\left(y_i^j \mid X_j, \theta_c\right) \tag{6}$$

Where $\mathcal{L}_d$ and $\mathcal{L}_c$ denotes the objective of the detection module and the correction module, respectively. θ_d and θ_c denote the parameters of the detection module and correction module. The linear combination of $\mathcal{L}_d$ and $\mathcal{L}_c$ is the overall training objective of the SM model.

$$\mathcal{L} = \lambda \cdot \mathcal{L}_c + (1-\lambda) \cdot \mathcal{L}_d \tag{7}$$

Where λ is a hyper-parameter and $\lambda \in (0,1)$.

3.2 Sparsity Regularization

The output of the detection network is a probability vector $P = (p_1, p_2, \cdots p_n)$. In the input of the CSC, the number of incorrect characters is usually far less than correct characters. Therefore, the probability vector P should be sparse and sharp. Accordingly, we attempt to study the sparsity regularization method for the CSC task. Specifically, we add a sparsity regularization item to the objective function as a penalty term to constraint the distribution of probability vector P.

$$\mathcal{L}' = \lambda \cdot \mathcal{L}_c + (1-\lambda) \cdot \mathcal{L}_d + \beta \cdot \sum_{j=1}^{N} R(P^j) \tag{8}$$

in which $R(P^j)$ is the sparsity regularization item, β is the hyper-parameter which balances the sparsity regularization term and the log-likelihood function. In this paper, we study two sparsity regularization methods: L_1 regularization and minimum entropy regularization.

L_1 regularization L_0 regularization and L_1 regularization are the most commonly used regularization method to control the sparsity of parameters. The return of L_0 regularization is positively correlated with the number of non-zero elements in the input, so minimizing the L_0 regularization term can make the parameters sparse. But L_0 regularization is a non-convex function, which means it is difficult to optimize. Therefore, researchers usually use L_1 regularization instead of L_0 regularization. For the probability vector P^j, L_1 regularization item is:

$$L_1(P^j) = \sum_{i=1}^{n} \left|p_i^j\right| \tag{9}$$

Where p_i^j is a probability and $p_i^j \in (0,1)$. Since the optimization strategy in this paper is to maximize the objective function $\mathcal{L}'$, $R(P^j)$ based on the L_1 regularization in this paper is defined as:

$$R(P^j) = -\sum_{i=1}^{n} p_i^j \tag{10}$$

In the CSC task, maximizing the $R(P^j)$ based on L_1 regularization will cause all p_i^j to approach 0. For j-th sentence in the dataset, If the character at the k-th position is incorrect, $\mathcal{L}_d$ will constrain p_k^j to approach 1. So L_1 regularization can control the distribution of P^j close to sparse theoretically. However, since the L_1 regularization item expects all p_i^j close to 0, even if the character is incorrect. This means that L_1 regularization cannot contribute to the probability vector P^j to be sharp.

We aim to contribute to the probability vectors P^j to be sparse and sharp. Accordingly, we introduce the minimum entropy approach.

Minimum entropy regularization In a statistical learning community, Minimum entropy is used less frequently than maximum entropy and L_1 regularization, there is also has been used as a regularization item. The literature (Grandvalet & Bengio, 2005) uses minimum entropy regularization to help the classifier based on semi-supervised learning make full use of beneficial unlabeled data, aiming to improve the robustness of the classifier. The literature (Zhang et al., 2018) uses the minimum entropy regularization item to contribute the distribution of attention vectors to sparse and sharp.

In the CSC task, we argue that P^j should be sparse and sharp. That is, the probability vector P^j should have low entropy. We use the minimum entropy regularization to meet this prior as follows.

$$\mathrm{Ent}(P^j) = -\sum_{i=1}^{n} p_i^j \log\left(p_i^j\right) \tag{11}$$

$$R(P^j) = -\mathrm{Ent}(P^j) = \sum_{i=1}^{n} p_i^j \log\left(p_i^j\right) \tag{12}$$

The derivative of the minimum entropy function $f(x) = x\log(x)$ is

$$f'(x) = 1 + \log(x) \tag{13}$$

When $x \in (0, 1/e)$, $f'(x) < 0$, and when $x \in (1/e, 1)$, $f'(x) > 0$, and $f(0) = f(1) = 0$. Therefore, when $x \in (0,1)$, maximizing the minimum entropy function will make the value of x to 0 or 1. For the CSC task, the value of the probability vector output by the detection module is preferably 0 or 1, corresponding to the location of the correct character and incorrect character. In summary, minimum entropy regularization can contribute to the CSC task at the theoretical level.

4 Experiment

In this section, we analyzed the experimental results of our method on the CSC task in detail and compared it with the results of baselines based on different methods. We also discussed how to choose model parameters and verified the robustness of our method.

"Id":	"A2-0023-1"
"Original_text":	"下个星期，我跟我朋唷打算去法国玩儿。"
"Wrong_ids":	" [9] "
"Correct_text":	"下个星期，我跟我朋友打算去法国玩儿。"

Table 1: Data sample.

Train and Val Data	Line
(Wang et al., 2018)	271,329
SIGHAN13	700
SIGHAN14	1301
SIGHAN15	970
Total	274,300
Test Data	**Line (Line with error character)**
SIGHAN13	1000(971)
SIGHAN14	1062(529)
SIGHAN15	1100(550)
Total	3162(2074)

Table 2: The statistic of the data.

4.1 Dataset

Train and Validation set In this paper, the training data is composed of (Wu et al., 2013), (Yu et al., 2014), and (Tseng et al., 2015). Following the literature (Wang et al., 2019), we add an additional 271K samples to our training data, which are provided by the literature (Wang et al., 2018). To observe the training process in real-time, we randomly selected 10% of these training data as the validation set and the other sample as the training set. We show one data sample in Table 1.

As shown in Table 1, a sample consists of four parts: Id, Original_text, Wrong_ids, and Correct_text. The Id is the id of the sample. The original_text denotes the input of the CSC task with\without incorrect characters. The Wrong_ids point out the location of the incorrect character, and it is an empty list when there are without incorrect characters in the input. The Correct_text denotes the corrected sentence, and it is the same as the Original_text when there are without incorrect characters in the input.

Test set Following the literature (Wang et al., 2019), we used the test dataset from the SIGHAN13 (Wu et al., 2013), SIGHAN14 (Yu et al., 2014), and SIGHAN15 (Tseng et al., 2015) benchmarks. Like related work, we also use OpenCC[1] to convert the characters in the test set from traditional Chinese to simplified Chinese. The statistic of the data is listed in Table 2.

[1] https://github.com/BYVoid/OpenCC

4.2 Evaluation Metrics

For this paper, we report precision (P), recall (R), and F1 scores as evaluation metrics, which are used by almost all CSC tasks. We report the sentence-level metrics on the detection sub-task and correction sub-task, i.e., we consider an input sentence has been correctly only when the output sentences of the model are completely consistent with our expected. As shown in **Formula** 24 through 26, the metrics are measured with the help of the confusion matrix which is shown in Table 3 following the literature (Tseng et al., 2015).

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) \tag{14}$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \tag{15}$$

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

For the CSC task, the better precision means that the fewer correct characters are recognized as incorrect characters, and the better recall means that the more incorrect characters can be detected/corrected.

4.3 Baselines

We compare our method to the following baselines.

FASpell[2] (Hong et al., 2019): The FASpell model uses the pre-trained model BERT as a denoising autoencoder (DAE) and decoder and uses confidence to filter candidate modifications instead of confusion set.

DCSpell (Li et al., 2021): The DCSpell model proposes a cloze-style detector-corrector framework that firstly detects whether a character is erroneous before correcting it. DCSpell uses the pre-trained discriminator ELECTRA (Clark et al., 2016) as a detection module and uses BERT to correct the incorrect characters.

BERT-Finetune (BERT-FT) (Devlin et al., 2019): Add a Softmax layer after the last layer of the BERT model, and use the training set to fine-tune the BERT model so that it can complete the CSC task.

Soft-Masked BERT[3] (Zhang et al., 2020): The SM model uses the Bi-GRU as the detection module, and then rewrites the input embedding according to the results of the detection module, and then enters it into the BERT model to complete the error correction.

Confusion matrix		**System Result**	
		Positive (Erroneous)	Negative (Correct)
Gold Standard	Positive	**TP**	**FN**
	Negative	**FP**	**TN**

Table 3: Confusion matrix.

4.4 Experimental Setting

In order to show the improvement of our method on the performance of the SM model, we use the same experimental setting as the literature (Zhang et al., 2020). Our code is based on BERT[4]. Following the literature (Zhang et al., 2020), when the fine-tuning process, we kept the default hyper-parameters and only used Adam. We did not use the dynamic learning rate strategy. The learning rate is set to $2e^{-5}$, the hidden dimension of Bi-GRU is set to 256, the batch size is set to 320, and the training epochs are set to 100. In order to save time, we use early stop strategy. We stop the training process early when the validation loss does not decrease for ten consecutive epochs. The value of λ is set to 0.8, which is the best value reported by (Zhang et al., 2020)

4.5 Result analysis

We list the experimental results of our method and four baseline models on the test set. It can be seen from Table 4, the precision and F1 score of the SM model with sparsity regularization method are better than baselines including SM model. Such experimental results show the effectiveness of the sparsity regularization method. In addition, we found that both the detection and correction recall of the SM model are better than that of the BERT-FT model. Such experimental results show that the Bi-GRU model is more effective than the BERT model as the detection module, which further supports the conclusion of literature (Zhang et al., 2020) and (Hong et al., 2019) that the error detection capability of BERT is poor.

Compared with the L_1 regularization, the minimum entropy regularization is better. This result is consistent with the analysis in Section 3.2. The L_1 regularization only can contribute to the sparsity of the probability vector output by the detection module, while minimum entropy regularization can contribute to sparsity and

[2] https://github.com/iqiyi/FASPell

[3] https://github.com/gitabtion/SoftMaskedBert-PyTorch

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

Test set	Method	Detection			Correction		
		P	R	F1	P	R	F1
SIGHAN	FASpell	36.4	43.3	39.6	30.1	35.7	32.7
	DCSpell	62.0	54.8	58.2	57.9	51.1	54.3
	BERT-FT	84.9	60.4	70.6	84.3	57.8	68.4
	SM	84.0	61.5	71.0	83.4	58.8	68.9
	SM + L_1	85.5	61.4	71.5	84.9	58.8	69.5
	SM + Entropy	**86.1**	**62.6**	**72.5**	**85.6**	**60.1**	**70.6**

Table 4: Performance of different methods on test set.

Method	Pe%	Detection			Correction		
		P	R	F1	P	R	F1
SM	20%	77.6	61.5	68.6	76.8	58.8	66.6
	40%	72.9	61.5	66.7	72.0	58.8	64.7
	60%	67.9	61.5	64.5	66.9	58.8	62.6
	80%	63.7	61.5	62.5	62.6	58.8	60.6
	100%	59.7	61.5	60.6	58.6	58.7	58.7
SM + L_1	20%	78.8	61.4	69.0	78.0	58.8	67.1
	40%	73.5	61.4	66.9	72.7	58.8	65.0
	60%	68.6	61.4	64.8	67.7	58.8	63.0
	80%	64.4	61.4	62.9	63.4	58.8	61.0
	100%	60.5	61.4	60.9	59.4	58.8	59.1
SM + Entropy	20%	80.4	62.6	70.4	79.8	60.1	68.5
	40%	75.0	62.6	68.2	74.2	60.1	66.4
	60%	70.3	62.6	66.2	69.4	60.1	64.4
	80%	66.1	62.6	64.3	65.2	60.1	62.5
	100%	62.5	62.6	62.5	61.5	60.1	60.8

Table 5: The result of robustness study.

sharpness. So, it is reasonable that the minimum entropy regularization method is better than L_1 regularization.

Further analysis of the experimental results shows that the detection precision and recall of the SM model with the minimum entropy regularization are better than the original SM model. This finding suggests that the minimum entropy regularization method can help the SM model fewer recognize correct characters as incorrect characters, and help it detect more incorrect characters. This result means that the minimum entropy regularization method that we have proposed therefore assists in improving the detection ability of the CSC model thoroughly. It should be noted that our method does not need any change in model architecture, that is, our method is theoretically applicable to all Chinese spelling error detection (CSED) tasks and the CSC tasks that include error detection sub-task.

In addition, the minimum entropy regularization improves the F1 score of error detection more than that of error correction. This result shows that the SM model with the minimum entropy regularization can detect and correct some sentences that the original SM model detects the incorrect characters but failed to correct. This result may be explained by the fact that the improvement of minimum entropy regularization for error correction comes from the advance of error detection. It can also make the probability vector output by the detection module sparse and sharp. According to this result, we can infer that our method is more suitable for the CSC model using soft strategy than the CSC model using fixed threshold strategy.

4.6 Robustness Study

Few input sentences contain incorrect characters in practical applications, such as text editing and correction. So, the robustness of the CSC model is very important when there are more and more without incorrect characters sentences in the test set. Therefore, we tested the robustness of our method.

We use the following methods to increase the number of sentences in the test set that without incorrect characters:

1) Randomly select Pe% of having incorrect characters samples from the test set;

Pe%	0%	20%	40%	60%	80%	100%
L_1	0.6	0.5	0.3	0.4	0.4	0.4
Entropy	1.7	1.9	1.7	1.8	1.9	2.1

Table 6: Effect of regularization item.

Method	β	Detection	Correction
		F1	F1
L_1	0.02	70.5	68.7
	0.04	**71.5**	**69.5**
	0.06	70.8	69.0
	0.08	70.9	69.1

Table 7: Effect of β (L_1 regularization).

Method	β	Detection	Correction
		F1	F1
L_1	0.02	72.1	70.1
	0.04	70.5	68.7
	0.06	**72.5**	**70.6**
	0.08	71.2	69.2

Table 8: Effect of β (minimum entropy regularization).

2) For each selected sample, use the Correct_text of the data to replace its Original_text, thereby generating a new test without incorrect characters sample;
3) Add all the new samples generated in step 2) to the test set.

We tested the performance of the SM model and our method when Pe% is 20%/40%/60%/80%/100%, and the results are reported in Table 5. During the robustness study, we will not fine-tune.

When Pe% takes different values, the recall of the three models always remains static. Because we just added some new without incorrect character samples to the test set. Therefore, the containing incorrect characters sample that the CSC model can detect and correct will not change while without fine-tune. So, the recall will not change.

As shown in Table 5, when the number of without incorrect characters samples in the test set increases, the precision of the three models decreases. However, minimum entropy regularization and L_1 regularization can still improve the performance of the CSC model. We study the improvement of the F1 score of the two kinds of sparsity regularization methods when Pe% takes different values.

Table 6 reports the result, it can be seen that the L_1 regularization can continually improve the performance of the CSC model when Pe% takes different values, but the improvement is slight. The minimum entropy regularization is better than the L_1 regularization, and as the value of Pe% increases, the minimum entropy regularization also shows a gradual improvement in the model performance. Therefore, it can be concluded that the two kinds of sparsity regularization studied in this paper are robust when the number of without incorrect characters samples in the test set increases.

4.7 Effect of Hyper Parameter β

We use the hyperparameter β to balance the objective function and the sparsity regularization item. The value of β is determined by the correcting F1 score on the test set. According to the experimental results, the best value of β is 0.04 when we use L_1 regularization, and the best value of β is 0.06 when we use minimum entropy regularization. The detailed experimental results are reported in Table 7 and Table 8.

5 Conclusion

This paper proposes to use the sparse regularization method to improve the performance of the CSC model. Specifically, we studied two kinds of sparsity regularization methods: L_1 regularization and minimum entropy regularization. By adding a sparsity regularization item to the objective function, the output of the detection module is close to sparse and sharp, so as to improve the performance of the CSC model. Although we only conducted experiments on the SM model, it should be noted that our method does not need any change in model architecture. Therefore, our method can apply to all Chinese spelling error detection (CSED) tasks and the CSC tasks that include the sub-task theoretical level. Experiments on the SIGHAN test set show that our method can effectively improve the performance of the CSC model, especially the minimum entropy regularization method.

References

Yu, J., & Li, Z. (2014). Chinese spelling error detection and correction based on language model, pronunciation, and shape. Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing.

Duan Jianyong, Y. Y., Wang Hao. (2021). Chinese Spelling Correction Method Based on Transformer Local Information and Syntax Enhancement Architecture. Acta Scientiarum Naturalium

Universitatis Pekinensis, 57(1), 61-67. https://doi.org/10.13209/j.0479-8023.2020.081.

Hládek, D., Staš, J., & Pleva, M. (2020). Survey of automatic spelling correction. Electronics, 9(10), 1670.

Tseng, Y.-H., Lee, L.-H., Chang, L.-P., & Chen, H.-H. (2015). Introduction to SIGHAN 2015 bake-off for Chinese spelling check. Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing.

Zhang, J., Zhao, Y., Li, H., & Zong, C. (2018). Attention with sparsity regularization for neural machine translation and summarization. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 27(3), 507-518.

Hirst, G., & Budanitsky, A. (2005). Correcting real-word spelling errors by restoring lexical cohesion. Natural Language Engineering, 11(1), 87-111.

Huang, Q., Huang, P., Zhang, X., Xie, W., Hong, K., Chen, B., & Huang, L. (2014). Chinese spelling check system based on tri-gram model. Proceedings of the third CIPS-SIGHAN joint conference on Chinese language processing.

Yeh, J.-F., Chang, L.-T., Liu, C.-Y., & Hsu, T.-W. (2017). Chinese spelling check based on N-gram and string matching algorithm. Proceedings of the 4th Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2017).

Duan, J., Wang, B., Tan, Z., Wei, X., & Wang, H. (2019). Chinese spelling check via bidirectional lstm-crf. 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC).

Wang, H., Wang, B., Duan, J., & Zhang, J. (2021). Chinese Spelling Error Detection Using a Fusion Lattice LSTM. Transactions on Asian and Low-Resource Language Information Processing, 20(2), 1-11.

Wang, D., Tay, Y., & Zhong, L. (2019). Confusionset-guided pointer networks for Chinese spelling check. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

Tan, M., Chen, D., Li, Z., & Wang, P. (2020). Spelling Error Correction with BERT based on Character-Phonetic. 2020 IEEE 6th International Conference on Computer and Communications (ICCC).

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT (1).

Cheng, X., Xu, W., Chen, K., Jiang, S., Wang, F., Wang, T., Chu, W., & Qi, Y. (2020). SpellGCN: Incorporating Phonological and Visual Similarities into Language Models for Chinese Spelling Check. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

Zhang, S., Huang, H., Liu, J., & Li, H. (2020). Spelling Error Correction with Soft-Masked BERT. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

Li, J., Yin, D., Wang, H., & Wang, Y. (2021). DCSpell: A Detector-Corrector Framework for Chinese Spelling Error Correction. Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval.

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2016). ELECTRA: PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS. ELECTRA, 85, 90.

Hong, Y., Yu, X., He, N., Liu, N., & Liu, J. (2019). Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm. Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019).

Grandvalet, Y., & Bengio, Y. (2005). Semi-supervised learning by entropy minimization. CAP, 367, 281-296.

Wu, S.-H., Liu, C.-L., & Lee, L.-H. (2013). Chinese spelling check evaluation at SIGHAN bake-off 2013. Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing.

Yu, L.-C., Lee, L.-H., Tseng, Y.-H., & Chen, H.-H. (2014). Overview of SIGHAN 2014 bake-off for Chinese spelling check. Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing.

Wang, D., Song, Y., Li, J., Han, J., & Zhang, H. (2018, oct" "nov). A Hybrid Approach to Automatic Corpus Generation for Chinese Spelling Check.Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing Brussels, Belgium.