

# Bi-DCSpell: A Bi-directional Detector-Corrector Interactive Framework for Chinese Spelling Check

Haiming Wu<sup>1</sup>, Hanqing Zhang<sup>2</sup>, Richeng Xuan<sup>3</sup> and Dawei Song<sup>1\*</sup>

<sup>1</sup>School of Computer Science & Technology, Beijing Institute of Technology, Beijing, China

<sup>2</sup>Institute of Computer Application Technology,

China North Industries Group Corporation Limited, Beijing, China

<sup>3</sup>Beijing Academy of Artificial Intelligence (BAAI), Beijing, China

{wuhm, dwsong}@bit.edu.cn, zhanghanqing.fly@gmail.com, rcxuan@baai.ac.cn

## Abstract

Chinese Spelling Check (CSC) aims to detect and correct potentially misspelled characters in Chinese sentences. Naturally, it involves the detection and correction subtasks, which interact with each other dynamically. Such interactions are bi-directional, i.e., the detection result would help reduce the risk of over-correction and under-correction while the knowledge learnt from correction would help prevent false detection. Current CSC approaches are of two types: correction-only or single-directional detection-to-correction interactive frameworks. Nonetheless, they overlook the bi-directional interactions between detection and correction. This paper aims to fill the gap by proposing a **Bi-directional Detector-Corrector** framework for CSC (*Bi-DCSpell*). Notably, *Bi-DCSpell* contains separate detection and correction encoders, followed by a novel interactive learning module facilitating bi-directional feature interactions between detection and correction to improve each other's representation learning. Extensive experimental results demonstrate a robust correction performance of *Bi-DCSpell* on widely used benchmarking datasets while possessing a satisfactory detection ability.

## 1 Introduction

Chinese Spelling Check (CSC) aims to automatically detect and correct spelling errors in Chinese sentences (Yu and Li, 2014; Wang et al., 2018; Huang et al., 2021). It serves as a foundation for a wide range of downstream applications in information retrieval (IR) and natural language processing (NLP), including Search Query Correction, Optical Character Recognition (OCR), and Essay Scoring. In recent years, CSC has garnered increasing attention from both academia and industry, emerging as a crucial task in the realm of Artificial Intelligence (Zhang et al., 2020; Li et al., 2024).

\*Corresponding authors

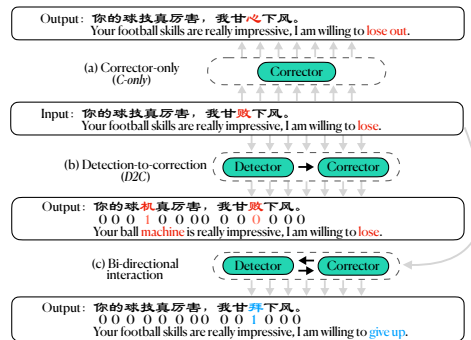


Figure 1: Comparison of three types of interactions: ‘correction-only’, ‘detection-to-correction’ and ‘bi-directional interactions’. The **wrongly/ground-truth** corrected characters in the candidates are in **red/blue**.

Most existing approaches to CSC can be categorized into two types of framework: (a) correction-only (*C-only*) and (b) detection-to-correction (*D2C*). The *C-only* does not contain a detection module. Instead, it directly adopts a specially designed corrector to perform the correction task (Hong et al., 2019; Zhang et al., 2020; Cheng et al., 2020; Li et al., 2021; Guo et al., 2021). The most recent *C-only* models are inspired by masked language model (MLM) (Devlin et al., 2019), where each character to be corrected is predicted using contextual information. The second type of framework possesses both detection and correction modules but the interaction between them is single-directional in a way akin to a pipeline (Cheng et al., 2020; Zhang et al., 2021; Zhu et al., 2022). That is, the detector first identifies the positions of errors, which are then used as prior knowledge for the correction module. While these two types of framework have been the dominant paradigms in the current CSC literature, they neglect the bi-directional interactions between detector and corrector. This oversight leads to issues such as the substitution of correct characters with incorrect ones (*over-correction*) or the failure to correct mis-

spelled characters (*under-correction*).

Figure 1 presents an example to illustrate the differences among *C-only*, *D2C*, and bi-directional interactions. The input “你的球机真厉害，我甘败下风。” contains two misspelled characters “机(jī, machine)” and “败(bài, fail)”, which should be corrected as “技(jì, skill)” and “拜(bài, worship)”, respectively. Figure 1(a) illustrates the *C-only* approaches that are typically underpinned by an MLM. Note that MLM tends to rectify a correct low-frequency collocation into a high-frequency one (Yang and Yu, 2022). In this example, as “败(bài, fail)” is more frequent in natural language than “拜(bài, worship)”, it is difficult for a *C-only* approach to correct “败(bài, fail)” properly in case there is no prior error information from detection. In Figure 1(b), the *D2C* interaction first uses an error detector that determines “败(bài, fail)” is wrong. The result is then fed into the correction module, which then successfully corrects “败(bài, fail)” as “拜(bài, worship)”. However, the corrector still may not be able to correct the other misspelled character “机(jī, machine)” as it is not pre-detected by the detector. This reflects that one-directional interaction (from detection to correction) can still lead to under-correction or over-correction.

To fill this gap, we propose a novel **Bi-directional Detector-Corrector** interactive framework for **Spelling** check (*Bi-DCSpell*). The detection and correction subtasks dynamically interact with each other, implying that knowledge acquired from one can be transferred to enhance the representation learning of the other. This bi-directional exchange of knowledge facilitates mutual improvement in performance. A concrete case is depicted in Figure 1(c). Different from the *C-only* and *D2C* approaches, *Bi-DCSpell* not only uses the detection result (“败(bài, fail)” is wrong), for the corrector to revise “败(bài, fail)” as “拜(bài, worship)”, but also feeds the correction result back to the detector as contextual information to help identify that “机(jī, machine)” may also be wrong.

Specifically, our approach uses two independent encoders: one dedicated to detection and the other to correction, to capture the subtask-specific features from the input sequence. Subsequently, we introduce an interactive learning module, which comprises a series of bi-directional cross-attention layers to enable the dynamic interactions between the two subtasks. Finally, two task-specific classifiers are used to output the detection labels and generate the correction sequence. The performance of

our method is evaluated on three widely used CSC benchmark datasets: SIGHAN13, SIGHAN14, and SIGHAN15. The results show that *Bi-DCSpell* surpasses interaction-free and uni-directional interactive baselines by 3.0% and 1.8% respectively in correction F1 score. Moreover, our method compares favorably against the current state-of-the-art approaches on all three datasets. A further ablation study demonstrates the effect of incorporating the bi-directional interactions in CSC. We also present a case study with concrete examples to illustrate the advantages of our model. The source code can be accessed at <https://github.com/haiming-wu/bi-spell>.

## 2 Related Work

### 2.1 Correction-only Approaches for CSC

Correction-only (*C-only*) is a kind of approaches in CSC, employing a corrector to directly correct the input sequence without error detection. Recent approaches along this line with a masked language model (MLM), and have achieved significant performance improvements (Hong et al., 2019; Wang et al., 2019; Zhang et al., 2020, 2021; Huang et al., 2021; Zhu et al., 2022). For instance, SpellGCN (Cheng et al., 2020) integrates phonological and visual similarity information into character classifiers using a graph network, which then feeds the graph representation into MLM. Realise (Xu et al., 2021) employs three distinct feature networks to capture phonetic, graphemic and semantic features, ultimately passing the fused representation through MLM. These methods focus on constructing CSC features and feeding them into an MLM-based corrector. In contrast, FASpell (Hong et al., 2019) leverages phonological and visual similarity features to construct a filtering model, selects the most suitable candidate Chinese characters from a pre-trained Language Model (PLM). To better formalize this type of CSC methods, we denote the model parameters as  $\theta_c$ , and the correction inference degenerates to:

$$P(y_i^C | \mathbf{x}) = P(y_i^C | \mathbf{x}, \theta_c) \quad (1)$$

where,  $y_i^C$  represents the corrected character at the  $i$ -th position in the input sequence  $\mathbf{x}$ , with  $C$  denoting the correction sub-task identifier.

### 2.2 Detection-to-correction for CSC

Detection-to-correction (*D2C*) represents the second type of existing CSC approaches, which first

use an error detector to detect the position information of misspelled characters and then feed the detection results to the corrector. Most approaches employ a conventional multiple-stage pipeline approach, such as Soft-masked BERT (Zhang et al., 2020), MLM-phonetics (Zhang et al., 2021) and DR-CSC (Huang et al., 2023). Unlike these pipeline *D2C* approaches, MDCSpell (Zhu et al., 2022) uses parallel detection and correction feature representation modules, and the corrector receives the detector’s hidden states, thus, the inference in correction incorporates the feature from both detection and correction. Accordingly, *D2C* approaches can be formalized as:

$$\begin{aligned} P(y_i^D | \mathbf{x}) &= P(y_i^D | \mathbf{x}, \boldsymbol{\theta}_d) \\ P(y_i^C | \mathbf{x}) &= P(y_i^C | \mathbf{x}, \boldsymbol{\theta}_d, \boldsymbol{\theta}_c) \end{aligned} \quad (2)$$

where  $\boldsymbol{\theta}_d$  denotes the parameters of detection.

Similar to MDCSpell, our method also employs parallel detection and correction processes. However, a key distinction is that, our model facilitates a bi-directional interactive exchange of information between the detection and correction processes, i.e., the features learned from one subtask are utilized as prior knowledge for the other. We expect such bi-directional interactions would reinforce the representation learning for both subtasks.

### 3 Bi-directional Interactive Framework

#### 3.1 Problem Formulation

The Chinese Spelling Check (CSC) task aims to detect and correct the misspelled characters in a Chinese sentence. It involves two sub-tasks: detection and correction. Thus it becomes a multi-task learning problem. Formally, given an input textual sequence  $\mathbf{x} = (w_1, w_2, \dots, w_n)$ , where each Chinese character  $w_i$  is taken from a vocabulary  $\mathcal{V}$ , a CSC model needs to detect whether each character is erroneous (i.e., misspelled), so as to output a sequence of detection labels  $\mathbf{y}^D = (y_1^D, y_2^D, \dots, y_n^D)$  and generate the corresponding corrections  $\mathbf{y}^C = (y_1^C, y_2^C, \dots, y_n^C)$ , where  $y_i^D \in \{0, 1\}$ ,  $y_i^C \in \mathcal{V}$ .

**Bi-directional Interaction** In this paper, we hypothesize that the bi-directional interaction between detection and correction would benefit the performance improvement of two sub-tasks, i.e., alleviating the challenges of under-correction and over-correction present in current approaches. To achieve this, we have crafted an interactive learning module that facilitates bi-directional transfer

of hidden knowledge (in term of features) between detection and correction. As shown in Figure 1(c), when the system detects whether the character in input text is incorrect, the detection module integrates feature information from correction for the detection process. Simultaneously, the correction module incorporates detection features from the detection module to make corrections. Therefore, the incorporation of bi-directional interactions in CSC can be formalized as computing the following probability for  $y_i^D, y_i^C$ :

$$P(y_i^D, y_i^C | \mathbf{x}) = P(y_i^D, y_i^C | \mathbf{x}, \boldsymbol{\theta}_d, \boldsymbol{\theta}_c) \quad (3)$$

#### 3.2 Bi-DCSpell for CSC

We propose a *Bi-directional Detector-Corrector* interactive framework for Chinese spelling check, namely *Bi-DCSpell*. The framework comprises three components: an Encoder module, an Interactive Learning module, and a Classifier Layer. The overall structure is depicted in Figure 2.

##### 3.2.1 Detection and Correction Encoders

To capture the subtask-specific features, two separate encoders are used, one for detection and the other for correction. The input text is first mapped to a sequence of embedding vectors by the embedding layer Embedding. Then, *Bi-DCSpell* takes two separate encoders for detection and correction (Encoder<sup>D</sup> and Encoder<sup>C</sup>), formalized as:

$$\begin{aligned} \mathbf{X} &= \text{Embedding}(\mathbf{x}) \\ \mathbf{H}_r^D &= \text{Encoder}^D(\mathbf{X}) \\ \mathbf{H}_r^C &= \text{Encoder}^C(\mathbf{X}), \end{aligned} \quad (4)$$

where  $\mathbf{H}_r^D, \mathbf{H}_r^C$  denote detection-specific and correction-specific feature representations respectively, which are essentially the hidden states without any cross-task interaction. To facilitate the calculation in the interactive learning module, we keep  $\mathbf{H}_r^D$  and  $\mathbf{H}_r^C$  at the same size  $R^{n \times d_h}$ , where  $d_h$  is the dimension of hidden state.

##### 3.2.2 Interactive Learning

To model the bi-directional interactions between detection and correction tasks, we design an interactive learning module as shown in the middle part of Figure 2. It consists of a series of stacked bi-directional interaction layers similar to the Transformer (Vaswani et al., 2017), with each layer incorporating two task-specific cross-attention networks, two learnable control gates and a feed-forward network.

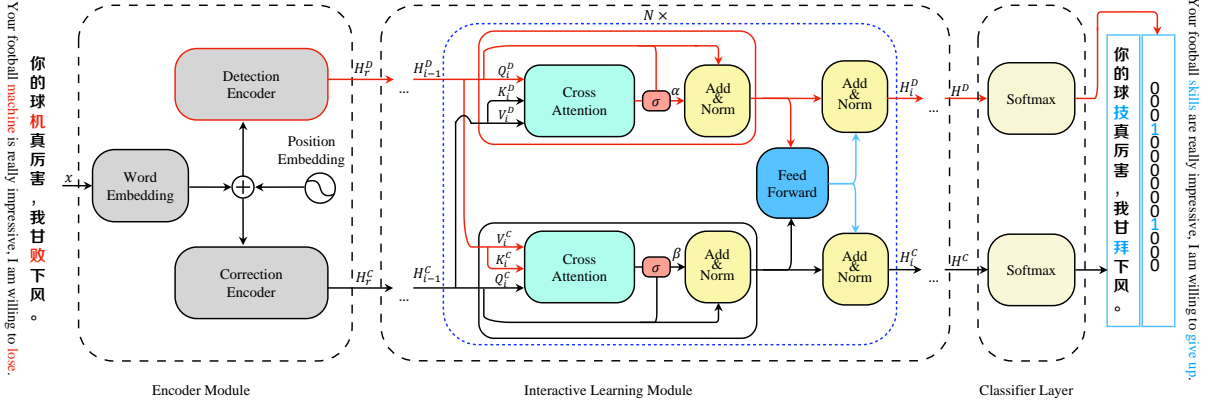


Figure 2: Overview of our *Bi-DCSpell* framework.  $\oplus$  is the matrix addition operation, and  $\sigma$  denotes learnable control gate.

**Task-specific Cross-attention Network.** Firstly, two task-specific cross-attention networks use the feature knowledge learnt from one task to inform the representation learning for the other, respectively. Here, we use the same structure as vanilla attention (Vaswani et al., 2017) in each cross-attention process, but the input involves two representation states (i.e., detection and correction) instead of one.

In detection cross-attention, the detection representation is updated by extracting corresponding information in correction features as follows:

$$\tilde{\mathbf{H}}_{is}^D = \text{softmax}\left(\frac{\mathbf{Q}_i^D \mathbf{K}_i^{D\top}}{\sqrt{d_K}}\right) \mathbf{V}_i^D \quad (5)$$

where,  $\tilde{\mathbf{H}}_{is}^D$  denotes the hidden state of interactive detection, and  $\text{softmax}$  is the  $\text{softmax}$  function, and  $\top$  denotes the transpose of matrix. The set of matrices (i.e., queries  $\mathbf{Q}_i^D$ , keys  $\mathbf{K}_i^D$ , and values  $\mathbf{V}_i^D$ ) is calculated by linear transformations from  $\mathbf{H}_{i-1}^D$  and  $\mathbf{H}_{i-1}^C$ . Specifically,  $\mathbf{H}_{i-1}^D$  is used to compute queries  $\mathbf{Q}_i^D$ , and  $\mathbf{H}_{i-1}^C$  is used to compute values  $\mathbf{V}_i^D$  and keys  $\mathbf{K}_i^D$ , where all the parameters in the linear functions are trainable.

And, similar to the calculation process of  $\tilde{\mathbf{H}}_{is}^D$ , we take  $\mathbf{H}_{i-1}^C$  and  $\mathbf{H}_{i-1}^D$  to compute queries  $\mathbf{Q}_i^C$ , keys  $\mathbf{K}_i^C$  and values  $\mathbf{V}_i^C$ , and then utilize the three matrices ( $\mathbf{Q}_i^C$ ,  $\mathbf{K}_i^C$ ,  $\mathbf{V}_i^C$ ) to update the correction representation by extracting the error position information in detection features  $\mathbf{H}_{i-1}^D$ .

$$\tilde{\mathbf{H}}_{is}^C = \text{softmax}\left(\frac{\mathbf{Q}_i^C \mathbf{K}_i^{C\top}}{\sqrt{d_K}}\right) \mathbf{V}_i^C \quad (6)$$

where  $\tilde{\mathbf{H}}_{is}^C$  denotes the hidden state of interactive correction.

**Learnable Control Gates** Then, considering that the degree of interaction between the correction and the detection is different for each example, we employ two learnable control gates ( $\alpha$  and  $\beta$ ) to regulate the extent of interaction between one sub-task and the other:

$$\begin{aligned} \alpha &= \sigma(W_i^D[\tilde{\mathbf{H}}_{is}^D, \mathbf{H}_{i-1}^D] + b_i^D) \\ \mathbf{H}_{is}^{D'} &= \text{LN}(\alpha \odot \tilde{\mathbf{H}}_{is}^D + (1 - \alpha) \odot \mathbf{H}_{i-1}^D) \\ \beta &= \sigma(W_i^C[\tilde{\mathbf{H}}_{is}^C, \mathbf{H}_{i-1}^C] + b_i^C) \\ \mathbf{H}_{is}^{C'} &= \text{LN}(\beta \odot \tilde{\mathbf{H}}_{is}^C + (1 - \beta) \odot \mathbf{H}_{i-1}^C) \end{aligned} \quad (7)$$

where,  $(W_i^*, b_i^*)$  are learnable gate parameters and  $\text{LN}$  denotes the layer normalization.  $\mathbf{H}_{is}^{D'}$  and  $\mathbf{H}_{is}^{C'}$  represent detection and correction hidden states after incorporating the correction/detection features in detection/correction representation learning.

**Feed-forward Network.** To merge the detection and correction feature knowledge, we designed a new feed-forward network inspired by Transformer. First, we concatenate the detection and correction hidden states and then feed it into a linear multiplication layer to learn the merged features.

$$\begin{aligned} \mathbf{H}_i^{DC} &= \mathbf{H}_{is}^{D'} \oplus \mathbf{H}_{is}^{C'} \\ \mathbf{H}_i^{DC'} &= \mathbf{W}_2[\text{ReLU}(\mathbf{W}_1 \mathbf{H}_i^{DC} + \mathbf{b}_1)] + \mathbf{b}_2 \end{aligned} \quad (8)$$

where  $\oplus$  denotes the tensor concatenation.  $\text{ReLU}$  is the  $\text{relu}$  activation function, and  $(\mathbf{W}_1, \mathbf{b}_1)$ ,  $(\mathbf{W}_2, \mathbf{b}_2)$  are two sets of parameters.

Finally, the merged hidden states  $\mathbf{H}_i^{DC'}$  are projected into a normalization function, and added to  $\mathbf{H}_{is}^{D'}$  and  $\mathbf{H}_{is}^{C'}$  to update the detection and correc-



tion representations, respectively.

$$\begin{aligned} \mathbf{H}_i^D &= LN(\mathbf{H}_{is}^{D'} + \mathbf{H}_i^{DC'}) \\ \mathbf{H}_i^C &= LN(\mathbf{H}_{is}^{C'} + \mathbf{H}_i^{DC'}) \end{aligned} \quad (9)$$

where  $\mathbf{H}_i^D$  and  $\mathbf{H}_i^C$  are the updated feature representations. In the last bi-directional interaction layer  $L$ , the updated representations  $\mathbf{H}_L^D$  and  $\mathbf{H}_L^C$  are utilized as the ultimate states  $\mathbf{H}^D$  and  $\mathbf{H}^C$ .

### 3.2.3 Classifier Layer

Hidden states  $\mathbf{H}^D$  and  $\mathbf{H}^C$  are fed into two task-specific classifiers to predict the detection label and generate the correction character for each corresponding character. Given the original input sentence  $\mathbf{x}$  and its two representation ( $\mathbf{H}^D$ ,  $\mathbf{H}^C$ ), we can compute its probability distributions  $p(\hat{y}_i^D|\mathbf{x})$  and  $\hat{y}_i^C$  for  $i$ -th character.

$$\begin{aligned} \hat{y}_i^D &= \text{softmax}(\mathbf{W}_D \mathbf{h}_i^D + \mathbf{b}_D) \\ \hat{y}_i^C &= \text{softmax}(\mathbf{W}_C \mathbf{h}_i^C + \mathbf{b}_C) \end{aligned} \quad (10)$$

where  $(\mathbf{W}_D, \mathbf{b}_D)$  and  $(\mathbf{W}_C, \mathbf{b}_C)$  are trainable parameters. Thus, the  $\hat{y}_i^D$  and  $\hat{y}_i^C$  denote output distributions of detection and correction, respectively.

### 3.2.4 Model Training

Given a set of manually labeled training data  $\{(\mathbf{x}_j, \mathbf{y}_j^D, \mathbf{y}_j^C)\}_{j=1}^m$  ( $m$  denotes the number of training samples), we use cross-entropy loss as objective functions to train the above model. To unify the training process, a linear multi-objective optimization function is designed to guide the learning of the model to balance the detection and correction processes.

$$\begin{aligned} \mathcal{L}^D &= - \sum_{j=1}^m \sum_{i=1}^n y_i^D \log(\hat{y}_i^D) \\ \mathcal{L}^C &= - \sum_{j=1}^m \sum_{i=1}^n y_i^C \log(\hat{y}_i^C) \\ \mathcal{L} &= \lambda \mathcal{L}^C + (1 - \lambda) \mathcal{L}^D \end{aligned} \quad (11)$$

where, the  $\lambda \in [0, 1]$  is a transfer-balanced hyper-parameter to guide model learning.

## 4 Experiments

### 4.1 Datasets and Pre-Processing

Extensive empirical evaluation is carried out on three widely used CSC datasets: SIGHAN13 (Wu et al., 2013), SIGHAN14 (Yu et al., 2014) and SIGHAN15 (Tseng et al., 2015). We train *Bi-DCSpell* using four datasets, three of which are training data sets from SIGHAN, with approximately 10,000 data samples. The fourth dataset is an additional set of training data generated by an

automatic method (Wang et al., 2018) with 271,009 samples.

Consistent with the current literature (Xu et al., 2021; Zhu et al., 2022), we use three test datasets from SIGHAN13, SIGHAN14 and SIGHAN15 to evaluate the trained model. Furthermore, we use the same pre-processing procedure and transform the characters in these datasets to simplified Chinese with OpenCC<sup>1</sup>.

## 5 Evaluation Metrics

The general evaluation metrics could be divided into sentence-level and character-level metrics. Specifically, a prediction at the sentence-level is only deemed correct if all of the misspelled characters in the sentence have been detected or corrected. Compared with character-level evaluation, sentence-level evaluation is more strict and tends to yield lower scores. Following the common practice in the literature (Hong et al., 2019; Cheng et al., 2020), we adopt the commonly used sentence-level precision, recall, and F1 score measures.

### 5.1 Implementation details

We use PyTorch (Paszke et al., 2019) to implement the proposed *Bi-DCSpell* the Transformers library (Wolf et al., 2020). For scientific comparison with existing methods, two pre-trained language models, Chinese-BERT-wwm (Cui et al., 2021) (abbreviated as BERT) and ChineseBERT (Sun et al., 2021), are used to initialize the embedding layer and the correction encoder, each model has 12 transformer layers with 12 attention heads and outputs for each token a hidden representation of dimensionality 768. We initialize the weights of the detector encoder using the Transformer parameters of the bottom two layers in the pre-trained language model. For model training, AdamW (Loshchilov and Hutter, 2018) is used as an optimizer with max epochs 20, the learning rate is set as 5e-5, and the batch size is set to 32. The training process takes about 10 hours on a single RTX A6000 (48GB GPU memory).

### 5.2 Model Settings

Considering that *Bi-DCSpell* is essentially a fine-tuning model for CSC, a wide range of CSC methods based on fine-tuning are selected as comparison models<sup>2</sup>: *PLM-FT* is a fine-tuned PLM on the

<sup>1</sup><https://github.com/BYVoid/OpenCC>

<sup>2</sup>Note that, due to the inherent disparity between a fine-tuned model and one trained through a combination of pre-

Dataset	Baseline	Backbone	Detection			Correction		
			Prec.	Rec.	F1	Prec.	Rec.	F1
SIGHAN13	<i>Soft-Masked BERT</i> * (Zhang et al., 2020)	BERT	81.1	75.7	78.3	75.1	70.1	72.5
	<i>SpellGCN</i> (Cheng et al., 2020)	BERT	80.1	74.4	77.2	78.3	72.7	75.4
	<i>MLM-phonetics</i> (Huang et al., 2021)	BERT	82.0	78.3	80.1	79.5	77.0	78.2
	<i>DCN</i> (Wang et al., 2021)	BERT	86.8	79.6	83.0	84.7	77.7	81.0
	<i>GAD</i> (Guo et al., 2021)	BERT	85.7	79.5	82.5	84.9	78.7	81.6
	<i>MDCSpell</i> (Zhu et al., 2022)	BERT	<b>89.2</b>	78.3	83.4	87.5	76.8	81.8
	<i>DORM</i> (Liang et al., 2023)	ChineseBERT	87.9	83.7	85.8	86.8	82.7	84.7
	<i>DR-CSC</i> (Huang et al., 2023)	ChineseBERT	88.5	83.7	86.0	87.7	83.0	85.3
	PLM-FT (ChineseBERT)	ChineseBERT	86.9	82.0	84.4	85.6	80.7	83.1
	<i>Bi-DCSpell</i> (BERT)	BERT	88.2	80.6	84.2	86.8	78.7	82.6
<i>Bi-DCSpell</i> (ChineseBERT)	ChineseBERT	89.0	<b>85.1</b>	<b>87.0</b>	<b>87.8</b>	<b>84.1</b>	<b>85.9</b>	
SIGHAN14	<i>Soft-Masked BERT</i> * (Zhang et al., 2020)	BERT	65.2	70.4	67.7	63.7	68.7	66.1
	<i>SpellGCN</i> (Cheng et al., 2020)	BERT	65.1	69.5	67.2	63.1	67.2	65.3
	<i>MLM-phonetics</i> (Zhang et al., 2021)	BERT	66.2	<b>73.8</b>	69.8	64.2	<b>73.8</b>	68.7
	<i>DCN</i> (Wang et al., 2021)	BERT	67.4	70.4	68.9	65.8	68.7	67.2
	<i>GAD</i> (Guo et al., 2021)	BERT	66.6	71.8	69.1	65.0	70.1	67.5
	<i>MDCSpell</i> (Zhu et al., 2022)	BERT	70.2	68.8	69.5	69.0	67.7	68.3
	<i>DORM</i> (Liang et al., 2023)	ChineseBERT	69.5	73.1	71.2	68.4	71.9	70.1
	<i>DR-CSC</i> (Huang et al., 2023)	ChineseBERT	70.2	73.3	71.7	69.3	72.3	70.7
	PLM-FT (ChineseBERT)	ChineseBERT	66.7	70.4	68.5	65.0	68.7	66.8
	<i>Bi-DCSpell</i> (BERT)	BERT	69.9	70.9	70.4	68.5	68.7	68.6
<i>Bi-DCSpell</i> (ChineseBERT)	ChineseBERT	<b>70.9</b>	73.7	<b>72.3</b>	<b>69.6</b>	72.4	<b>71.0</b>	
SIGHAN15	<i>Soft-Masked BERT</i> * (Zhang et al., 2020)	BERT	67.6	78.7	72.7	63.4	73.9	68.3
	<i>SpellGCN</i> (Cheng et al., 2020)	BERT	74.8	80.7	77.7	72.1	77.7	75.9
	<i>MLM-phonetics</i> (Huang et al., 2021)	BERT	77.5	83.1	80.2	74.9	80.2	77.5
	<i>DCN</i> (Wang et al., 2021)	BERT	77.1	80.9	79.0	74.5	78.2	76.3
	<i>GAD</i> (Guo et al., 2021)	BERT	75.6	80.4	77.9	73.2	77.8	75.4
	<i>MDCSpell</i> (Zhu et al., 2022)	BERT	80.8	80.6	80.7	78.4	78.2	78.3
	<i>CoSPA</i> (Yang and Yu, 2022)	ChineseBERT	79.0	82.4	80.7	76.7	80.0	78.3
	<i>DORM</i> (Liang et al., 2023)	ChineseBERT	77.9	84.3	81.0	76.6	82.8	79.6
	<i>DR-CSC</i> (Huang et al., 2023)	ChineseBERT	<b>82.9</b>	84.8	83.8	<b>80.3</b>	82.3	81.3
	PLM-FT (ChineseBERT)	ChineseBERT	79.3	83.0	81.1	77.2	80.7	78.9
<i>Bi-DCSpell</i> (BERT)	BERT	79.6	82.4	81.0	77.5	80.2	78.8	
<i>Bi-DCSpell</i> (ChineseBERT)	ChineseBERT	82.6	<b>85.4</b>	<b>84.0</b>	80.2	<b>84.1</b>	<b>82.1</b>	

Table 1: Experimental results on SIGHAN13, SIGHAN14 and SIGHAN15 test sets. Each model includes sentence-level precision, recall, and F1 score for both detection and correction. \*Due to the incompatibility of character-level results in the original paper (Zhang et al., 2020), the results for *Soft-Masked BERT* in the table are sourced from (Zhang et al., 2021), maintaining consistency in training data and metrics with our method. We evaluated *Bi-DCSpell* using two backbones, Chinese-BERT-wwm (BERT) and ChineseBERT for a fair comparison.

training data, which is indeed a standard *C-only* model and can be viewed as a simplified version of *Bi-DCSpell* without the interactive learning module and the detection encoder. *Soft-Masked BERT* (Zhang et al., 2020), *MLM-phonetics* (Zhang et al., 2021) and *DR-CSC* (Huang et al., 2023) utilize a pipeline model, they all use the information of the previous step as the prior information of the next step. *SpellGCN* (Cheng et al., 2020) incorporates phonological and visual similarity knowledge representation into BERT by employing a specialized graph convolutional network. *DCN* (Wang et al., 2021) uses a dynamically connected network to measure the degree of dependence between any two adjacent Chinese characters. *CoSPA* (Yang and Yu, 2022) reports an alterable copy mechanism to in-

training and fine-tuning, we do not choose the models relying on pre-training for comparison, such as the current state-of-the-art CSC model *PTCSpell* (Wei et al., 2023)

crease the generation probability of the original input, thereby mitigating the over-correction. *DORM* (Liang et al., 2023) allows the direct interaction between textual and phonetic information. *MDCSpell* (Zhu et al., 2022) utilizes multi-task learning to jointly model detection and correction, implementing a multi-task learning model with task-specific feature encoders similar to those in *Bi-DCSpell*. However, it neglects the knowledge interaction between detection and correction.

### 5.3 Main Results

The experimental results are shown in Table 1. Note that the performance of *Bi-DCSpell* is influenced by the hyper-parameter  $\lambda$ , which balances detection and correction objectives in the loss function. The best performing results are reported in Table 1, and a detailed discussion of the perfor-

Method	Interaction	Det-F1	Cor-F1
PLM-FT	no	81.1	78.9
D2C	uni-directional	82.7 <sub>+1.6</sub>	80.7 <sub>+1.8</sub>
<i>Bi-DCSpell</i>	bi-directional	84.0 <sub>+2.9</sub>	81.9 <sub>+3.0</sub>

Table 2: The results of three interaction frameworks on the SIGHAN15 test set. D2C is the Bi-DCSpell with the fixed gate  $\alpha = 0.0$ .

mance across different hyper-parameter settings can be found in Section 5.4.

Overall, our proposed *Bi-DCSpell* consistently achieves the best F1 score for both detection and correction on all three SIGHAN datasets. This result indicates the effectiveness of our method.

In the same scenario (i.e., backbone is ChineseBERT), compared with *PLM-FT*, which is a *C-only* baseline, *Bi-DCSpell* significantly improves the correction F1 score by 2.8%, 4.2%, and 3.0% respectively, validating the necessity of modeling the interactions between detection and correction.

Compared with BERT-based *Soft-Masked BERT*, *MLM-phonetics*, *MDCSpell*, *CoSPA* and ChineseBERT-based *DR-CSC*, which all adopt uni-directional interaction framework (i.e., *D2C*), *Bi-DCSpell* performs significantly better on all the datasets in both scenarios, confirming that modeling bi-directional interactions between detection and correction can further improve the CSC performance.

Compared with *MLM-phonetics*, *CoSPA* and *SpellGCN* with BERT backbone, which all explicitly introduce extra phonological and/or visual information into the inference process, the proposed *Bi-DCSpell* (BERT) does not incorporate any external information but it still achieves better F1 scores on almost all datasets. This result demonstrates the competitive performance and robustness of our proposed method.

Compared with *DORM* that modeled the direct interaction between textual and phonetic information in the same scenario, *Bi-DCSpell* (ChineseBERT) achieves better F1 scores on all datasets.

## 5.4 Ablation Study

In this subsection, we use SIGHAN15 test sets to examine the effects of various core components and parameters of our model, including the interactive learning module, the number of bi-directional interaction layers, the control gate factors  $\alpha$  and  $\beta$ , and the balancing hyperparameter  $\lambda$ .

**Bi-directional interactive learning.** Table 2 shows a result comparison between three differ-

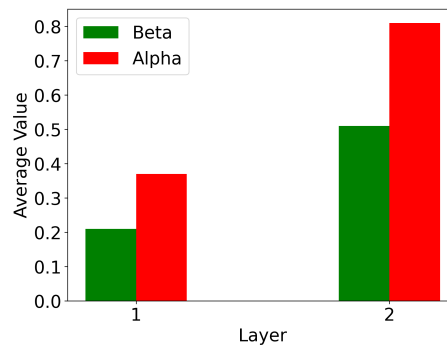


Figure 3: The average value of  $\alpha$  and  $\beta$  with different bi-directional interaction layers.

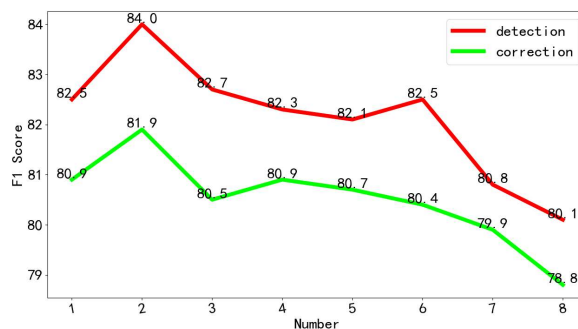


Figure 4: The F1 scores for detection and correction vary as the number of bi-directional interaction layers increases.

ent frameworks, including *PLM-FT* (*C-only*), *D2C* and *Bi-DCSpell*. The results illustrate that incorporating the interactions between detection and correction lead to significant performance improvements, and the modeling of bi-directional interactions brings more benefits than considering uni-directional interactions only.

### The range of optimal degree of interaction.

The *Bi-DCSpell* framework contains two learnable controlling gates:  $\alpha$  and  $\beta$ . We compute the mean values of  $\alpha$  and  $\beta$  across different interaction layers in the optimal model scenario, as illustrated in Figure 3. In both layers,  $\alpha$  and  $\beta$  are less than 1, indicating a moderate level of bi-directional interaction. And the information exchange from correction to detection is also lower than from detection to correction. Specifically, in the first layer, both parameters reflect a low degree of interaction, while in the second layer, the bi-directional interaction becomes more pronounced.

**The number of bi-directional interaction layers.** As shown in Figure 4, the F1 scores of detection and correction are subject to some fluctuations when the number of bi-directional interaction layers changes between 1 and 8. These fluctuations

case 1	Input	团长将士兵 <b>布(bù)</b> 署(shù)在城外, 让他们 <b>安(ān)</b> 兵不动。
	<i>C-only</i>	团长将士兵 <b>布置(bù zhì)</b> 在城外, 让他们 <b>按(àn)</b> 兵不动。
	<i>D2C</i>	0 0 0 0 0 0 <b>0</b> 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 团长将士兵 <b>布(bù)</b> 署(shù)在城外, 让他们 <b>按(àn)</b> 兵不动。
	<i>Bi-DCSpell</i>	0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 团长将士兵 <b>部(bù)</b> 署(shù)在城外, 让他们 <b>按(àn)</b> 兵不动。
	Translation	The captain deployed the soldiers outside the city and ordered them to stand still.
case 2	Input	任何 <b>因(yīn)</b> 难(nán)都不能 <b>下(xià)</b> 倒(dǎo)有坚强意志的 <b>对(duì)</b> 员(yuán)们(men)。
	<i>C-only</i>	任何 <b>困难(kùn nán)</b> 都不能 <b>击败(jī bài)</b> 有坚强意志的 <b>队(duì)</b> 员(yuán)们(men)。
	<i>D2C</i>	0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 任何 <b>原因(yuán yīn)</b> 都不能 <b>下(xià)</b> 倒(dǎo)有坚强意志的 <b>队(duì)</b> 员(yuán)们(men)。
	<i>Bi-DCSpell</i>	0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 任何 <b>困难(kùn nán)</b> 都不能 <b>吓(xià)</b> 倒(dǎo)有坚强意志的 <b>队(duì)</b> 员(yuán)们(men)。
	Translation	No difficulty can intimidate team members with strong determination.

Table 3: Examples of CSC results from *Bi-DCSpell*, in comparison with results from *C-only* and *D2C* baselines. Red and blue are used to mark incorrect and correct characters, respectively.

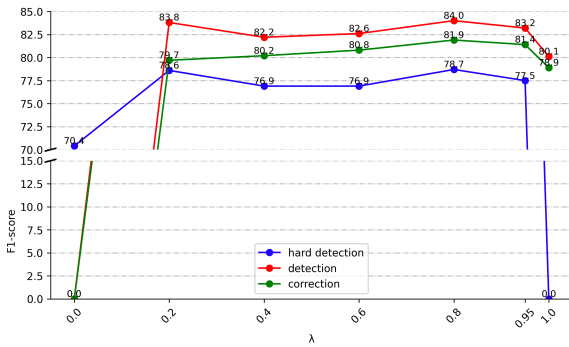


Figure 5: The F1 scores with respect to  $\lambda$ .

generally increase when the number varies from 1 to 2, and then show a downward trend. Especially when the number is 2, the F1 scores reach a maximum value in both detection and correction. Therefore, considering the efficiency, we choose a layer number of 2 for the remaining experiments.

**The balancing hyperparameter  $\lambda$  in loss function.** Figure 5 provides the detection, correction and hard detection<sup>3</sup> F1 scores while  $\lambda$  increases. The result shows that the detection and correction F1 scores are 0.0 when  $\lambda$  is 0 (where the correction classifier is not trained), and the hard detection F1 score is 0.0 when  $\lambda$  is 1 (where the detection classifier is not trained). Importantly, the hard detection rapidly increases when  $\lambda$  increases from 0 to 0.2. When the  $\lambda$  goes from 0.8 to 1.0, the detection and correction performance are both decreasing. When the  $\lambda = 0.8$ , the correction F1 score reaches a maximum. Thus, we set the  $\lambda$  as 0.8 for an optimum performance of correction.

<sup>3</sup>To clarify, the evaluation of detection performance is based on the corrected sentences generated by the corrector. The evaluation of ‘hard detection’ is based directly on the results of the detection classifier layer.

## 5.5 Case Study

To further illustrate the effectiveness of *Bi-DCSpell* in comparison with the *C-only* and *D2C* baselines, we analyze two concrete cases in Table 3.

In the first case, the MLM-based *C-only* model mis-corrected “布署(bù shǔ)” into “布置(bù zhì)”, a high-frequency collocation, resulting in over-correction. On the other hand, the *D2C* model did not correct “布署(bù shǔ)”, due to its failure in detection. This is an example of under-correction. In contrast, the *Bi-DCSpell* not only correctly detected “布署(bù shǔ)” and “安(ān)” as misspelled characters, but also made accurate corrections.

The second case involves three discrete misspellings (“困难(yīn nán), 下倒(xià dǎo) and 对员(duì yuán)”). The *C-only* model correctly corrected two misspelled characters, but mis-corrected “下倒(xià dǎo)” to a more common phrase “击败(jī bài)” in this context, resulting in an over-correction. The *D2C* model was able to address some of the issues. However, its detection module failed to identify the wrong character “下(xià)”, and as a consequence its correction module was not able to correct it. Additionally, the detection module mis-identified the “难(nán)” in “困难(yīn nán)” as incorrect character, and in turn its correction module considered “任何(rèn hé)” and “原因(yuán yīn)” as a better match, leading to the mis-correction of “困难(yīn nán)” to “原因(yuán yīn)”. On the other hand, *Bi-DCSpell* successfully identified all misspelled characters and accurately corrected them, showing a better ability to tackle both over-correction and under-correction problems.



Method	D-F1	C-F1	C-I	S-I
PLM-FT	81.1	78.9	95.8	41.9
<i>Bi-DCSpell</i>	84.0	82.1	97.7	45.7

Table 4: The consistent output statistics. The ‘D-F1’, ‘C-F1’, ‘C-I’ and ‘S-I’ denote metric ‘detection f1-score’, ‘correction f1-score’, ‘character-level consistency’ and ‘sentence-level consistency’.

## 6 Conclusions

In this paper, we propose a novel CSC framework, namely *Bi-DCSpell*, which constitutes a bi-directional interactive detector-corrector framework, mutually enhancing the feature representation for the detection and correction sub-tasks. Comprehensive experiments and empirical analyses attest to the effectiveness of our approach. The explicit modeling of bi-directional interactions between sub-tasks also holds potential significance for analogous tasks, like grammatical error correction, which deserves further research in the future.

## 7 Limitations

### 7.1 Inconsistent Output

Due to the output layer of the detector-corrector framework having two results: detection labels and correction sentences, which occupy different distribution spaces, the issue of inconsistent outputs poses a new challenge. In *Bi-DCSpell*, two cross-attention networks are utilized to perform feature selection from the corrector to the detector and from the detector to the corrector. During model training, the *Bi-DCSpell* will dynamically adjust the two tasks from joint training. And we conducted a new comparison using two consistency metrics: character-level and sentence-level. Specifically, we compare PLM-FT (ChineseBERT) and *Bi-DCSpell* (ChineseBERT) on the SIGHAN15 dataset. The results are as Table 4:

From the above table, compared with PLM-FT (ChineseBERT), which is a fine-tuned PLM with two classifiers for corrector and detector, respectively, *Bi-DCSpell* achieves better consistency scores at both the character-level and sentence-level.

### 7.2 Language Limitation

In this work, we focus only on the spelling check of Chinese characters, because CSC is very different from other languages such as English. Specifically, (1) there are no delimiters between words. (2) Chinese have more than 100,000 characters,

and about 3,500 are frequently used in daily life, and most characters have similar visual and/or similar pronunciations. However, we believe that the bi-directional interactions between detection and correction is also important for spelling check in English texts, which is worth an in-depth investigation in the future.

### 7.3 Running Efficiency

In our code implementation, we have not paid too much attention to the running efficiency of the proposed methods. More precisely, it is expensive to take about 10 hours on an RTX A6000 (48GB GPU memory) to finish the training process. We think that the training efficiency can be improved by deploying the model training process on multiple GPUs and using data-parallel operations to increase the training batch size and shorten the training time.

## 8 Acknowledgements

This work is funded by the Natural Science Foundation of China (grant no: 62376027), National Science and Technology Major Project(2022ZD0116306), and Beijing Municipal Natural Science Foundation (grant no: 4222036 and IS23061).

## References

- Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. Spellgc: Incorporating phonological and visual similarities into language models for chinese spelling check. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 871–881.
- Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Zhao Guo, Yuan Ni, Keqiang Wang, Wei Zhu, and Guotong Xie. 2021. Global attention decoder for Chinese spelling error correction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1419–1428.

- 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.
- Haojing Huang, Jingheng Ye, Qingyu Zhou, Yinghui Li, Yangning Li, Feng Zhou, and Hai-Tao Zheng. 2023. [A frustratingly easy plug-and-play detection-and-reasoning module for Chinese spelling check](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11514–11525, Singapore.
- Li Huang, Junjie Li, Weiwei Jiang, Zhiyu Zhang, Minchuan Chen, Shaojun Wang, and Jing Xiao. 2021. Phmospell: Phonological and morphological knowledge guided chinese spelling check. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5958–5967.
- Chengzhang Li, Ming Zhang, Xuejun Zhang, and Yonghong Yan. 2024. Mcrspell: A metric learning of correct representation for chinese spelling correction. *Expert Systems with Applications*, 237:121513.
- Jing Li, Gaosheng Wu, Dafei Yin, Haozhao Wang, and Yonggang Wang. 2021. Dcspell: A detector-corrector framework for chinese spelling error correction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1870–1874.
- Zihong Liang, Xiaojun Quan, and Qifan Wang. 2023. [Disentangled phonetic representation for Chinese spelling correction](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13509–13521, Toronto, Canada.
- Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.
- Zijun Sun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2065–2075.
- Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. Introduction to sighthan 2015 bake-off for chinese spelling check. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 32–37.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- 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*, pages 2437–2446.
- 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.
- 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.
- Xiao Wei, Jianbao Huang, Hang Yu, and Qian Liu. 2023. [PTCSpell: Pre-trained corrector based on character shape and Pinyin for Chinese spelling correction](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6330–6343, Toronto, Canada. Association for Computational Linguistics.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.
- Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013. Chinese spelling check evaluation at sighthan bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42.
- Heng-Da Xu, Zhongli Li, Qingyu Zhou, Chao Li, Zizhen Wang, Yunbo Cao, He-Yan Huang, and Xian-Ling Mao. 2021. Read, listen, and see: Leveraging multimodal information helps chinese spell checking. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 716–728.
- Shoujian Yang and Lian Yu. 2022. Cospa: An improved masked language model with copy mechanism for chinese spelling correction. In *Uncertainty in Artificial Intelligence*, pages 2225–2234. PMLR.
- Junjie Yu and Zhenghua Li. 2014. Chinese spelling error detection and correction based on language model, pronunciation, and shape. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 220–223.

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

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuo-huan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. 2021. Correcting chinese spelling errors with phonetic pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2250–2261.

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

Chenxi Zhu, Ziqiang Ying, Boyu Zhang, and Feng Mao. 2022. Mdcspell: A multi-task detector-corrector framework for chinese spelling correction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1244–1253.

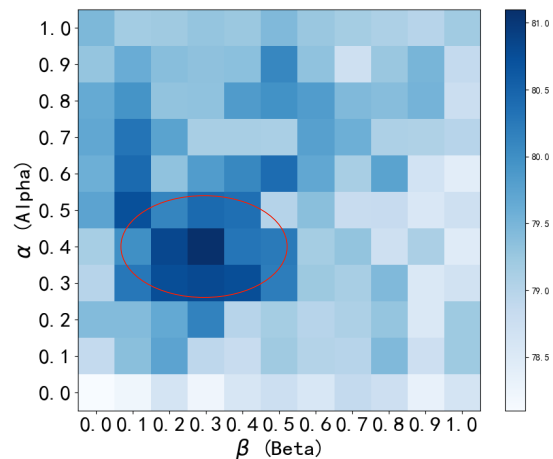


Figure 6: Correction F1 scores of Bi-DCSpell with different combinations of interaction degrees ( $\alpha$  and  $\beta$ ). The deeper color indicates the higher performance.

### A Optimal interaction range with hyper-parameter selection

To verify the effectiveness of the learnable control gates  $\alpha$  and  $\beta$  described in Section 4.4, we further adopted a hyper-parameter selection strategy to identify their optimal ranges. We use a heatmap to visualize the impact of these two gate values (set them as parameters) on the correction F1 score, as depicted in Figure 6. In general, a moderate bi-directional interaction yield the highest correction F1 score, aligning with the findings in Figure 3, outperforming both unidirectional ( $\alpha = 0.0$  or  $\beta = 0.0$ ) and fully bi-directional interactions ( $\alpha = 1.0$  and  $\beta = 1.0$ ). Specifically, the model achieves the optimal correction F1 score (darkest color) at ( $\alpha = 0.4, \beta = 0.3$ ), with progressively lighter colors when moving away from this point. In other words, the F1 score gradually decreases, reaching a minimum at ( $\alpha = 0.0, \beta = 0.0$ ) where there is no interaction.