Disentangled Phonetic Representation for Chinese Spelling Correction

Zihong Liang[1], **Xiaojun Quan**[1]*, **Qifan Wang**[2]
[1]School of Computer Science and Engineering, Sun Yat-sen University [2]Meta AI
liangzh63@mail2.sysu.edu.cn, quanxj3@mail.sysu.edu.cn, wqfcr@fb.com

Abstract

Chinese Spelling Correction (CSC) aims to detect and correct erroneous characters in Chinese texts. Although efforts have been made to introduce phonetic information (Hanyu Pinyin) in this task, they typically merge phonetic representations with character representations, which tends to weaken the representation effect of normal texts. In this work, we propose to disentangle the two types of features to allow for direct interaction between textual and phonetic information. To learn useful phonetic representations, we introduce a pinyin-to-character objective to ask the model to predict the correct characters based solely on phonetic information, where a separation mask is imposed to disable attention from phonetic input to text. To avoid overfitting the phonetics, we further design a self-distillation module to ensure that semantic information plays a major role in the prediction. Extensive experiments on three CSC benchmarks demonstrate the superiority of our method in using phonetic information[1].

1 Introduction

Chinese Spelling Correction (CSC) is a task to detect and correct erroneous characters in Chinese sentences, which plays an indispensable role in many natural language processing (NLP) applications (Martins and Silva, 2004; Gao et al., 2010). Previous research (Liu et al., 2010) shows that the misuse of homophonic characters accounts for roughly 83% of the spelling errors. We present two such cases in Table 1. In the first one, the erroneous characters of "户秃" are difficult to be corrected by only literal text because the input sample is too short and the two characters are entirely unrelated to the semantic meaning of this sample. However, their pronunciation easily helps us associate them with the correct answer "糊涂" which shares the same pronunciation as "户秃". The second case exhibits a similar phenomenon but is more complicated as the model must distinguish between "记得" and "记的" further. These two examples illustrate that misspelled characters could be recognized and corrected with the introduction of phonetic information. In Mandarin Chinese, Hanyu Pinyin (shortened to *pinyin*) is the official romanization system for phonetic transcription. It uses three components of initials, finals, and tones to express the pronunciation and spelling of Chinese characters. As the pronunciation similarity of Chinese characters is primarily determined by their initial or final sounds rather than their tones, we focus solely on the initials and finals as the phonetic features of Chinese characters.

Source	可是我忘了，我真户秃(hu tu)。 But I forgot, I am so household bald.	
Target	可是我忘了，我真糊涂(hu tu)。 But I forgot, I am so silly.	
BERT	可是我忘了，我真护突。	✗
PinyinBERT	可是我忘了，我真糊涂。	✓
REALISE	可是我忘了，我真户涂。	✗
Our DORM	可是我忘了，我真糊涂。	✓
Source	可是现在我什么事都不济的(ji de)。 But I can't do anything right now.	
Target	可是现在我什么事都不记得(ji de)。 But I don't remember anything now.	
BERT	可是现在我什么事都不记得。	✓
PinyinBERT	可是现在我什么事都不记的。	✗
REALISE	可是现在我什么事都不记的。	✗
Our DORM	可是现在我什么事都不记得。	✓

Table 1: Two examples of Chinese Spelling Correction and the predictions by different models. Misspelled characters are highlighted in red and the corresponding answers are in blue. The phonetic transcription of key characters is bracketed. PinyinBERT is a special BERT model which takes as input only phonetic features without characters. REALISE is a state-of-the-art model.

As pre-trained language models like BERT (Devlin et al., 2019) have dominated various NLP tasks, researchers explore incorporating pinyin features

*Corresponding authors
[1]https://github.com/liangzh63/DORM-CSC

into pre-trained language models for the CSC task. There are mainly two approaches. First, the pinyin of a Chinese character is encoded and fused into the character representation with a gate mechanism (Wang et al., 2021; Huang et al., 2021; Xu et al., 2021; Zhang et al., 2021). Second, a pronunciation prediction objective is introduced to model the relationship among phonologically similar characters (Liu et al., 2021; Ji et al., 2021; Li et al., 2022a). Despite considerable performance gain, these methods suffer from two potential issues. First, pinyin information may be neglected or dominated by textual information during training because of the entanglement between pinyin and textual representations. As the first case shows in Table 1, a special BERT model taking only the pinyin sequence as input without Chinese characters can detect and correct the erroneous characters, while REALISE (Xu et al., 2021), which encodes and fuses textual and pinyin information with a gate mechanism, ignores one of the errors. Second, the introduction of pinyin features may weaken the representation of normal texts. Take the second case in Table 1 for example. While an ordinary BERT model can correct the misspelled character "的" in the input, REALISE fails to do that. This problem could be explained by the over-reliance of REALISE on or overfitting pinyin information.

Based on the above observations, we propose **D**isentangled ph**O**netic **R**epresentation **M**odel (DORM) for CSC. Our motivation is to decouple text and pinyin representations to allow for direct interaction between them to make better use of phonetic information. Specifically, we first construct a phonetics-aware input sequence by appending the pinyin sequence to the original textual input, where a common set of position embeddings is used to relate the two sub-sequences. In doing so, textual features are allowed to capture phonetic information as needed from the pinyin part during training and inference. Then, to learn useful pinyin representations, we introduce a pinyin-to-character prediction objective, where a separation mask is imposed to disallow attention from pinyin to text to ask the model to recover the correct characters only from pinyin information. The pinyin-to-character task is auxiliary during training and its prediction will be discarded at inference time.

Intuitively, pinyin should serve to complement but not replace textual information in CSC for two reasons. First, there is a one-to-many relation between pinyin and Chinese characters, and it is more difficult to recover the correct characters solely from pinyin than from Chinese characters. Second, pinyin representations are not pre-trained as textual representations in existing language models. Therefore, the model should avoid overly relying on pinyin which may cause overfitting. Inspired by deep mutual learning (Zhang et al., 2018) and self-distillation (Mobahi et al., 2020), we propose a self-distillation module to force the prediction of our model to be consistent with that when a raw-text input is supplied. To this end, KL-divergence is applied to the two sets of soft labels.

Experiments are conducted on three SIGHAN benchmarks and the results show that our model achieves substantial performance improvement over state-of-the-art models. Further analysis demonstrates that phonetic information is better utilized in our model. The contributions of this work are summarized threefold. First, we disentangle text and pinyin representations to allow for direct interaction between them. Second, we introduce a pinyin-to-character task to enhance phonetic representation learning with a separation mask imposed to disable attention from pinyin to text. Third, a self-distillation module is proposed to prevent over-reliance on phonetic features. Through this work, we demonstrate the merit of our approach to modeling pinyin information separately from the text.

2 Related Work

2.1 Chinese Spelling Correction

Chinese Spelling Correction has drawn increasing interest from NLP researchers. The current methodology of this task has been dominated by neural network-based models, especially pre-trained language models, and can be divided into two lines.

One line of work focuses on better semantic modeling of textual features (Hong et al., 2019; Guo et al., 2021; Li et al., 2022c). They treat CSC as a sequence labeling task and adopt pre-trained language models to acquire contextual representations. Soft-Masked BERT (Zhang et al., 2020) employs a detection network to predict whether a character is erroneous and then generates soft-masked embedding for the correction network to correct the error. MDCSpell (Zhu et al., 2022) is a multi-task detector-corrector framework that fuses representations from the detection and correction networks.

Another line of work is incorporating phonetic information into the task, motivated by the observation that the misuse of homophonic characters accounts for a large proportion of the errors (Liu et al., 2010). MLM-phonetics (Zhang et al., 2021) and PLOME (Liu et al., 2021) employ a word replacement strategy to replace randomly-selected characters with phonologically or visually similar ones in the pre-training stage. REALISE (Xu et al., 2021) and PHMOSpell (Huang et al., 2021) utilize multiple encoders to model textual, phonetic, and visual features and employ a selective gate mechanism to fuse them. SCOPE (Li et al., 2022a) imposes an auxiliary pronunciation prediction task and devises an iterative inference strategy to improve performances. However, these methods generally merge textual and phonetic features without direct and deep interaction between them, which may lead to ineffective use of phonetic information. By contrast, our method decouples the two types of features to learn isolated phonetic representations and use them to assist textual information for CSC.

2.2 Self-Distillation

Knowledge distillation (Hinton et al., 2015) is a technique that tries to distill a small student model from a large teacher model. As a special distillation strategy, deep mutual learning (Zhang et al., 2018) allows several student models to collaboratively learn and teach each other during training. Particularly, it is referred to as self-distillation (Mobahi et al., 2020) when the student models share the same parameters. Self-distillation has been applied in CSC and brings performance improvement. SDCL (Zhang et al., 2022) encodes both original and corresponding correct sentences respectively, and adopts contrastive loss to learn better contextual representations. CRASpell (Liu et al., 2022) constructs a noisy sample for each input and applies KL-divergence for the two outputs to improve the performance on multi-typo sentences. Our method differs from CRASpell in two aspects. First, one of our student models takes as input a phonetics-aware sequence with disentangled textual and phonetic representations. Second, the purpose of our self-distillation design is to reduce overfitting phonetic information when training the model.

3 Methodology

The motivation of our **D**isentangled ph**O**netic **R**epresentation **M**odel (DORM) for Chinese Spelling Correction (CSC) is to allow for direct and deep interaction between textual and phonetic features by decoupling Chinese character and pinyin representations. To enable effective pinyin representations, we introduce a pinyin-to-character objective that requires the model to restore the correct characters purely from pinyin information. Inspired by deep mutual learning (Zhang et al., 2018) and self-distillation (Mobahi et al., 2020), we further introduce a self-distillation module to prevent the model from overfitting pinyin information. In the following, we first formulate the task (§3.1) and then introduce DORM in detail (§3.2). Finally, we introduce how to pre-train the model for better textual and pinyin representations (§3.3).

3.1 Problem Definition

Given a Chinese sentence $X = \{x_1, x_2, .., x_n\}$ of n characters that may include erroneous characters, we use $Y = \{y_1, y_2, .., y_n\}$ to represent the corresponding correct sentence. The objective of CSC is to detect and correct the erroneous characters by generating a prediction $\hat{Y} = \{\hat{y}_1, \hat{y}_2, .., \hat{y}_n\}$ for the input X, where $\hat{y}_i$ is the character predicted for x_i. Apparently, the CSC task can be formulated as a sequence labeling task in which all the Chinese characters constitute the label set.

3.2 Architecture

As illustrated in Figure 1, our DORM consists of a phonetics-aware input sequence, a unified encoder with separation mask, a pinyin-to-character objective, and a self-distillation module. The phonetics-aware input is constructed by appending the pinyin sequence to the original textual input. The separation mask is imposed to disallow attention from pinyin to text to avoid information leaks. The pinyin-to-character objective is designed to learn useful phonetic representations. In the self-distillation module, the model conducts two forward passes with the phonetics-aware sequence and the raw text as input respectively to obtain two sets of distributions, and the difference between them is minimized by KL-divergence.

Phonetics-Aware Input Sequence The pinyin of each Chinese character is a sequence of the Latin alphabet and is composed of *initials*, *finals* and *tones* to denote the pronunciation. If characters share the same initial or final, their pronunciations are usually related or similar. In our method, we only consider initials and finals as pinyin information for CSC, as empirically tones are not related to this task. Given the

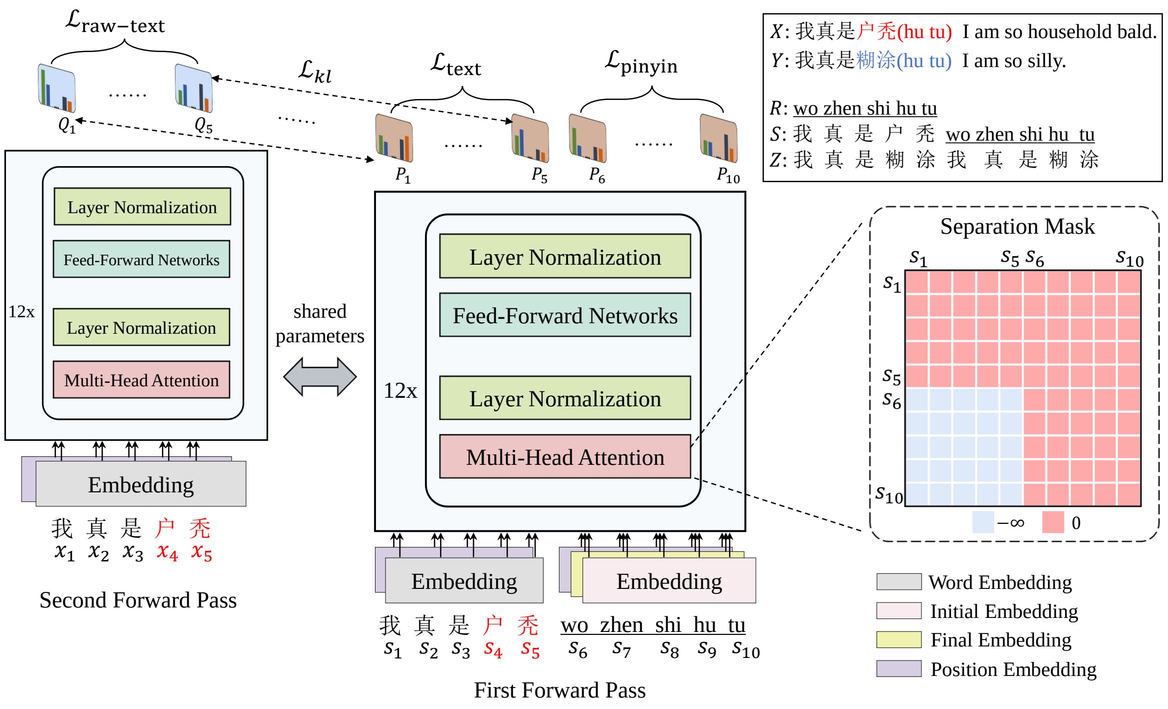


Figure 1: The architecture of the proposed DORM, which consists of a phonetics-aware input sequence S, an encoder with separation mask, a pinyin-to-character objective, and a self-distillation module. X is the original input sentence, R is the pinyin sequence of X, Y is the corresponding correct sentence, and Z is the prediction label based on S. Pinyin sequences are underlined to distinguish them from English sentences. Misspelled characters are shown in red and the corresponding correct characters are in blue. For self-distillation, the model conducts two forward passes with different inputs, and the output distributions are constrained by KL-divergence.

input X, we denote its pinyin sequence as $R = \{(\text{init}_1, \text{final}_1), (\text{init}_2, \text{final}_2), .., (\text{init}_n, \text{final}_n)\}$, where init_i and final_i are the initial and final of character x_i, respectively. Then, we append R to X and obtain a phonetics-aware sequence $S = \{s_1, s_2, .., s_n, s_{n+1}, s_{n+2}, .., s_{n+n}\}$ as the final input, where s_i is defined as follows.

$$s_i = \begin{cases} x_i, & 1 \le i \le n \\ \text{init}_{i-n}, \text{final}_{i-n}, & n+1 \le i \le n+n \end{cases}. \quad (1)$$

Encoder with Separation Mask We adopt BERT (Devlin et al., 2019) with a stack of 12 Transformer (Vaswani et al., 2017) blocks as our encoder. Each Chinese character is encoded as the sum of word embedding, position embedding, and segment embedding. Similarly, the pinyin of each character is encoded as the sum of initial embedding, final embedding, position embedding, and segment embedding, where the position embedding is the same as the character. As a result, the representations of the phonetics-aware input sequence S can be denoted by $H^0 = \{h_1^0, h_2^0, .., h_{n+n}^0\}$.

The contextual representation of each token is updated by aggregating information from other tokens via multi-head attention networks (MHA). In the l-th layer, the output O^l of each attention head is computed as:

$$\begin{aligned} Q^l, K^l, V^l &= H^{l-1}W_Q^{l\top}, H^{l-1}W_K^{l\top}, H^{l-1}W_V^{l\top}, \\ A^l &= \text{softmax}\left(\frac{Q^l K^{l\top}}{\sqrt{d}} + M\right), \quad (2) \\ O^l &= A^l V^l. \end{aligned}$$

where W_Q^l, W_K^l, W_V^l are trainable parameters, H^{l-1} is the output of the previous layer, d is the size of the dimension, and M is a mask matrix.

Specifically, we apply a separation mask to allow for attention from text representations to phonetic representations but not vice versa. Thus, we define the mask matrix $M \in \mathbb{R}^{2n \times 2n}$ in Eq. (2) as:

$$M_{ij} = \begin{cases} -\infty, & \text{if } n+1 \le i \le 2n \text{ and } 1 \le j \le n \\ 0, & \text{otherwise} \end{cases}. \quad (3)$$

The separation mask ensures that pinyin representations cannot gather information from textual characters when $M_{ij} = -\infty$. Next, O^l from all heads are concatenated then passed through a linear transformation network and a normalization network. After that, the resulting representations are fed into a feed-forward network followed by another normalization network to generate H^l.

The final contextual representations $H = \{h_1, h_2, .., h_{n+n}\}$ are produced by taking the last-layer hidden states of the encoder. Then, we compute the probability distribution for the i-th character based on h_i by:

$$P_i = \text{softmax}(E * h_i + b) \in \mathbb{R}^{|V|}. \tag{4}$$

where E is word embedding parameters, $|V|$ denotes the size of vocabulary, and b is a trainable parameter. The prediction loss for the textual part of S is computed as:

$$\mathcal{L}_{\text{text}} = \frac{1}{n}\sum_{i=1}^{n} -\log P(y_i|S). \tag{5}$$

Pinyin-to-Character Objective To design the auxiliary pinyin-to-character task, we make a copy of the gold output Y to obtain $Z = \{z_1, .., z_n, z_{n+1}, .., z_{n+n}\}$ as the prediction labels of S, where $z_1, .., z_n = y_1, .., y_n$ and $z_{n+1}, .., z_{n+n} = y_1, .., y_n$. The prediction loss of the pinyin part in S is defined as:

$$\mathcal{L}_{\text{pinyin}} = \frac{1}{n}\sum_{i=n+1}^{n+n} -\log P(z_i|S). \tag{6}$$

At inference time, we obtain the prediction $\hat{Y} = \{\hat{y}_1, ..\hat{y}_n, \hat{y}_{n+1}, .., \hat{y}_{n+n}\}$, where $\hat{y}_i = \text{argmax}(P_i)$. We discard the prediction for the pinyin part and use $\{\hat{y}_1, ..\hat{y}_n\}$ as the final output.

Self-Distillation Module After obtaining the output distribution for each character by Equation (4), the model conducts another forward pass with the original sequence X as input, giving rise to another output distribution $Q_i \in \mathbb{R}^{|V|}$ for each character x_i. The two sets of distributions are then forced to be close by applying bidirectional KL-divergence:

$$\mathcal{L}_{kl} = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(\mathcal{D}_{kl}(P_i||Q_i) + \mathcal{D}_{kl}(Q_i||P_i)). \tag{7}$$

Besides, the prediction objective of the second pass is also included in the training:

$$\mathcal{L}_{\text{raw-text}} = \frac{1}{n}\sum_{i=1}^{n} -\log P(y_i|X). \tag{8}$$

Joint Learning To train the model, we combine the phonetics-aware loss and the self-distillation loss into a joint training framework as:

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{text}} + \alpha\mathcal{L}_{\text{pinyin}}}_{\text{phonetics-aware loss}} + \underbrace{\beta\mathcal{L}_{kl} + \gamma\mathcal{L}_{\text{raw-text}}}_{\text{self-distillation loss}}. \tag{9}$$

where α, β, and γ are tunable hyperparameters.

3.3 Pre-training

Pinyin sequences can be regarded as a special form of natural language sequences. Since they are not presented in the original pre-training process of language models, reasonably, they can be pre-trained on large-scale corpora to obtain better pinyin representations for fine-tuning. Therefore, we pre-train DORM on two large corpora, namely wiki2019zh[2] and weixin-public-corpus[3]. The format of input sequences and the model structure are the same as in fine-tuning. DORM is trained by recovering 15% randomly selected characters in the input, which were replaced by phonologically similar or random characters. Moreover, the pinyin-to-character objective is also included. More implementation details are given in Appendix A.

4 Experiments

In this section, we introduce the details of our experiments to evaluate the proposed model.

4.1 Datasets and Metrics

We conduct main experiments on three CSC benchmarks, including SIGHAN13 (Wu et al., 2013), SIGHAN14 (Yu et al., 2014), and SIGHAN15 (Tseng et al., 2015). Following previous work (Wang et al., 2019; Cheng et al., 2020; Xu et al., 2021), we merge the three SIGHAN training sets and another 271K pseudo samples generated by ASR or OCR (Wang et al., 2018) as our training set. We evaluate our model on the test sets of SIGHAN13, SIGHAN14, and SIGHAN15, respectively. Since the original SIGHAN datasets are in Traditional Chinese, they are converted to Simplified Chinese by OpenCC[4]. We adopt the pypinyin toolkit[5] to obtain the pinyin of each character.

We use the metrics of sentence-level precision, recall, and F1 to evaluate our model for detection and correction. For detection, all misspelled characters in a sentence should be detected correctly to count it as correct. For correction, a sentence is considered as correct if and only if the model detects and corrects all erroneous characters in this sentence. More details about the datasets and the metrics are presented in Appendix B.

[2]https://github.com/brightmart/nlp_chinese_corpus
[3]https://github.com/nonamestreet/weixin_public_corpus
[4]https://github.com/BYVoid/OpenCC
[5]https://pypi.org/project/pypinyin/

Dataset	Methods	Detection (%)			Correction (%)		
		precision	recall	F1	precision	recall	F1
SIGHAN15	BERT	74.2	78.0	76.1	71.6	75.3	73.4
	SpellGCN (Cheng et al., 2020)	74.8	80.7	77.7	72.1	77.7	75.9
	DCN (Wang et al., 2021)	77.1	80.9	79.0	74.5	78.2	76.3
	PLOME (Liu et al., 2021)	77.4	81.5	79.4	75.3	79.3	77.2
	MLM-phonetics (Zhang et al., 2021)	77.5	83.1	80.2	74.9	80.2	77.5
	REALISE (Xu et al., 2021)	77.3	81.3	79.3	75.9	79.9	77.8
	LEAD (Li et al., 2022b)	**79.2**	82.8	80.9	**77.6**	81.2	79.3
	DORM (ours)	77.9	**84.3**	**81.0**	76.6	**82.8**	**79.6**
SIGHAN14	BERT	64.5	68.6	66.5	62.4	66.3	64.3
	SpellGCN (Cheng et al., 2020)	65.1	69.5	67.2	63.1	67.2	65.3
	DCN (Wang et al., 2021)	67.4	70.4	68.9	65.8	68.7	67.2
	MLM-phonetics (Zhang et al., 2021)	66.2	**73.8**	69.8	64.2	**73.8**	68.7
	REALISE (Xu et al., 2021)	67.8	71.5	69.6	66.3	70.0	68.1
	LEAD (Li et al., 2022b)	**70.7**	71.0	70.8	**69.3**	69.6	69.5
	DORM (ours)	69.5	73.1	**71.2**	68.4	71.9	**70.1**
SIGHAN13	BERT	85.0	77.0	80.8	83.0	75.2	78.9
	SpellGCN (Cheng et al., 2020)	80.1	74.4	77.2	78.3	72.7	75.4
	DCN (Wang et al., 2021)	86.8	79.6	83.0	84.7	77.7	81.0
	MLM-phonetics (Zhang et al., 2021)	82.0	78.3	80.1	79.5	77.0	78.2
	REALISE (Xu et al., 2021)	**88.6**	82.5	85.4	87.2	81.2	84.1
	LEAD (Li et al., 2022b)	88.3	83.4	85.8	**87.2**	82.4	84.7
	DORM (ours)	87.9	**83.7**	**85.8**	86.8	**82.7**	**84.7**

Table 2: Overall results of DORM and baselines on SIGHAN13/14/15 in detection/correction precision, recall, and F1. The best results are shown in bold and the second-best results are underlined. The results of baselines are cited from the corresponding papers.

4.2 Baselines

We compare our DORM with the following baselines. **BERT** (Devlin et al., 2019) is initialized with pre-trained BERT$_{\text{base}}$ and fine-tuned on the training set directly. **SpellGCN** (Cheng et al., 2020) models prior knowledge between phonetically or graphically similar characters with graph convolutional networks. **DCN** (Wang et al., 2021) uses a Pinyin Enhanced Candidate Generator to introduce phonological information and then models the connections between adjacent characters. **MLM-phonetics** (Zhang et al., 2021) integrates phonetic features during pre-training with a special masking strategy that replaces words with phonetically similar words. **PLOME** (Liu et al., 2021) utilizes GRU networks to model phonological and visual knowledge during pre-training with a confusion set-based masking strategy. **REALISE** (Xu et al., 2021) learns semantic, phonetic, and visual representations with three encoders and fuses them with a gate mechanism. **LEAD** (Li et al., 2022b) models phonetic, visual, and semantic information by a contrastive learning framework. Additionally, the implementation details of our DORM are presented in Appendix C.

4.3 Overall Results

As the overall results show in Table 2, the proposed DORM outperforms existing state-of-the-art methods in both detection and correction F1 scores on SIGHAN13/14/15 test datasets, which demonstrates the effectiveness of this model. Compared with other models utilizing phonetic and visual features (e.g., REALISE and PLOME) and models pre-trained on larger corpora (e.g., PLOME and MLM-phonetics), which have access to further external information, DORM still achieves favourable improvement in detection/correction F1. We also note that the improvements in detection/correction recall are prominent and consistent across different test sets. These results suggest that our model is able to capture phonetic information more effectively. Although the improvement in precision is not as encouraging as recall and F1, its performance is still competitive compared with other methods also including phonetic information in this task.

5 Analysis and Discussion

In this section, we further analyze and discuss our model quantitatively and qualitatively.

5.1 Ablation Study

To investigate the contribution of key components of our model, we ablate them in turn and report the F1 performance for the correction task on SIGHAN13/14/15 in Table 3. As shown in the first group, eliminating the separation mask leads to considerable performance declines, showing that

Method	Correction F1 (Δ)		
	SIGHAN13	SIGHAN14	SIGHAN15
DORM	**84.7**	**70.1**	**79.6**
w/o SM	83.6 (-1.1)	67.4 (-2.7)	79.0 (-0.6)
w/o SD	83.1 (-1.6)	69.1 (-1.0)	78.9 (-0.7)
w/o $\mathcal{L}_{\text{pinyin}}$	84.2 (-0.5)	68.3 (-1.8)	79.2 (-0.4)
w/o pre-training	83.7 (-1.0)	66.9 (-3.2)	78.6 (-1.0)
w/o SD&SM	82.1 (-2.6)	68.3 (-1.8)	77.1 (-2.5)
w/o SD&$\mathcal{L}_{\text{pinyin}}$	83.0 (-1.7)	68.7 (-1.4)	77.8 (-1.8)
w/o SD&$\mathcal{L}_{\text{pinyin}}$&SM	81.4 (-3.3)	67.3 (-2.8)	76.9 (-2.7)

Table 3: Results of ablation study in correction F1 on SIGHAN13/14/15, where "*w/o*" means without, "$\mathcal{L}_{\text{pinyin}}$" means the pinyin-to-character objective, "SM" denotes the separation mask, "SD" denotes the self-distillation module, and "Δ" denotes the change of performance.

preventing pinyin representations from attending to textual information is necessary to learn useful phonetic representations. Moreover, removing self-distillation also leads to performance degradation, which suggests that the module is useful to avoid overfitting pinyin. When $\mathcal{L}_{\text{pinyin}}$ is discarded, the performance drops correspondingly, meaning that phonetic features tend to be ignored without the pinyin-to-character objective. Moreover, a sharp decline is observed when dropping the pre-training phase, which implies that pre-training on large-scale corpora indeed improves phonetic representations. More experimental results of various combinations in the second group further reveal the contribution of these components.

5.2 Effect of Phonetic Knowledge

According to the assumption, more phonetically similar misspellings should be restored with the assistance of phonetic knowledge. To show this, we focus on the recall performance of different models on phonetically misspelled characters of SIGHAN13/14/15. We collect 1130/733/668 such misspellings from the three test sets, accounting for about 93%/95%/95% of all misspellings, respectively. From the results in Table 4, we can note that our model achieves 93.5%/82.1%/90.0% recall scores and outperforms two phonetic-based models (i.e., SCOPE (Li et al., 2022a) and REALISE) consistently. In particular, it beats BERT by a large margin. These results indicate that phonetic knowledge is essential to CSC and our model is able to utilize phonetic knowledge more effectively.

5.3 Effect of Self-Distillation

The self-distillation module is introduced for DORM to avoid overfitting pinyin information. To show the effect of this module, we record the number of normal characters that are mistakenly treated

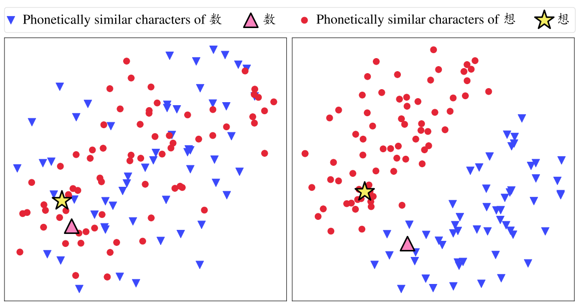
Model	Recall (%)		
	SIGHAN13	SIGHAN14	SIGHAN15
DORM	**93.5**	**82.1**	**90.0**
SCOPE†	91.6	80.2	87.6
REALISE†	89.8	78.2	84.7
BERT	88.8	75.2	82.8

Table 4: The performance of models in restoring phonetically misspelled characters on SIGHAN13/14/15. Results marked with "†" are obtained by executing released models from corresponding papers.

as misspellings (i.e., overcorrections), as well as the number of misspellings not restored (i.e., undercorrections) in the three test sets. The results in Table 5 show that the number of undercorrections is significantly reduced when phonological information but not self-distillation is introduced, while the number of overcorrections generally stays unchanged except on SIGHAN13. These results demonstrate that after including the self-distillation module, the numbers of overcorrections and undercorrections are both reduced compared with the baseline, demonstrating that self-distillation indeed alleviates the overfitting issue.

Model	#Overcorrections/#Undercorrections		
	SIGHAN13	SIGHAN14	SIGHAN15
BERT	103/129	175/177	120/106
DORM *w/o* SD	118/75	172/134	119/63
DORM	107/77	161/136	116/65

Table 5: The effect of self-distillation in reducing overcorrections and undercorrections on SIGHAN13/14/15. "*w/o* SD" means without the self-distillation module.

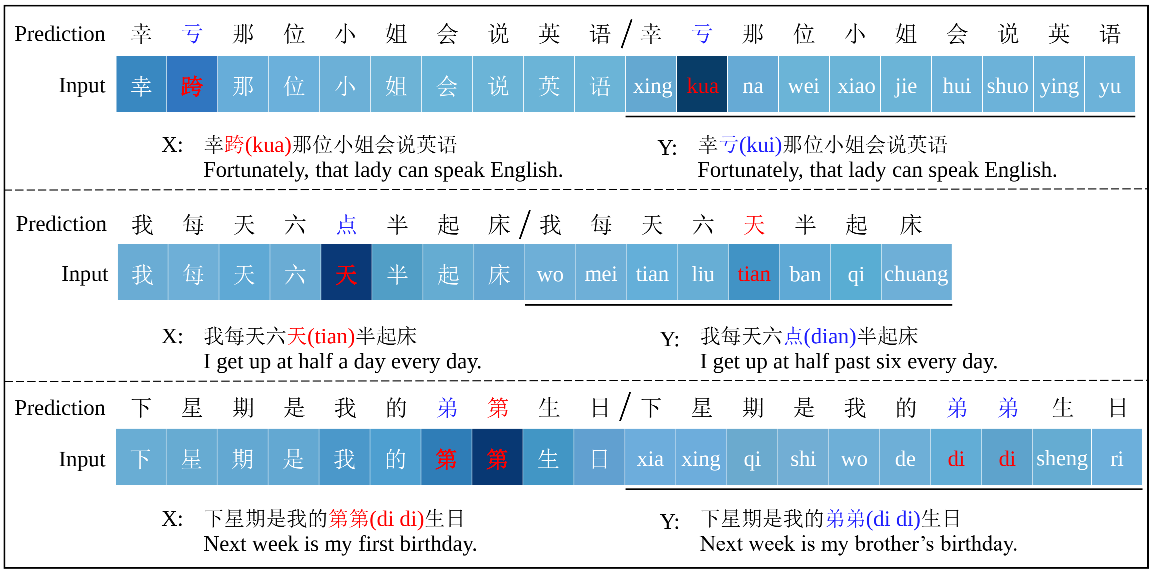


(a) BERT (b) DORM

Figure 2: Visualization of character representations, in which (a) is fine-tuned BERT and (b) is our DORM. Two pivot characters "数" (number) and "想" (want) have different pronunciations.

5.4 Visualization

Ideally, the introduction of phonetic knowledge should improve Chinese character representations


Prediction 幸 亏 那 位 小 姐 会 说 英 语 / 幸 亏 那 位 小 姐 会 说 英 语
Input 幸 跨 那 位 小 姐 会 说 英 语 xing kua na wei xiao jie hui shuo ying yu
Prediction 我 每 天 六 点 半 起 床 / 我 每 天 六 天 半 起 床
Input 我 每 天 六 天 半 起 床 wo mei tian liu tian ban qi chuang
Prediction 下 星 期 是 我 的 弟 第 生 日 / 下 星 期 是 我 的 弟 弟 生 日
Input 下 星 期 是 我 的 第 第 生 日 xia xing qi shi wo de di di sheng ri

X: 幸跨(kua)那位小姐会说英语
Fortunately, that lady can speak English.

Y: 幸亏(kui)那位小姐会说英语
Fortunately, that lady can speak English.

X: 我每天六天(tian)半起床
I get up at half a day every day.

Y: 我每天六点(dian)半起床
I get up at half past six every day.

X: 下星期是我的第第(di di)生日
Next week is my first birthday.

Y: 下星期是我的弟弟(di di)生日
Next week is my brother's birthday.

Figure 3: Case study on SIGHAN15, where misspellings and corresponding answers are highlighted in red and blue, respectively. The phonetic input is underlined and its prediction is discarded during inference. Attention weights from misspellings to the input sequence are also visualized where darker colors mean larger weights.

in that phonetically similar characters are pulled closer in the space. To show the effect, we employ t-SNE (van der Maaten and Hinton, 2008) to visualize character representations generated by our model, with fine-tuned BERT as the baseline. We randomly select two characters "数" and "想" of different pronunciations and collect about 60 phonetically similar characters provided by Wu et al. (2013) for eacknow. We plot the two groups of representations in Figure 2, from which we can note that the representations produced by fine-tuned BERT are scattered and less distinguishable between the groups. However, our model separates them into two distinct clusters according to the pivot characters, demonstrating that our model can better model the relationships among phonetically similar characters for CSC.

5.5 Case Study

Finally, we provide a case study with two good and one bad examples to analyze our model. We visualize the attention weights from each misspelled character to the other positions in the phonetics-aware sequence to show how our model utilizes phonetic information. As presented in Figure 3, in the first case both the textual and phonetic parts make correct predictions. After looking into the attention weights, we note the prediction for the misspelled position pays much attention to its previous position, the current position, and its pinyin position. In the second case, while the phonetic part leads to a wrong prediction, our model focuses more on the textual part and eventually makes a correct prediction. In the third case, although the prediction of the pinyin part is accurate, the textual part fails to pay much attention to it and causes a wrong prediction, suggesting that there is still room for improvement in balancing phonetic and semantic information. These cases intuitively show how our model uses phonetic information to correct misspelled characters.

6 Conclusion

In this paper, we propose DORM in an attempt to improve the effect of using phonetic knowledge in Chinese Spelling Correction (CSC). To this end, we propose to disentangle textual and phonetic features and construct a phonetics-aware input to allow for direct interaction between them. We also introduce a pinyin-to-character objective to force the model to recover the correct characters based solely on pinyin information, where a separation mask is applied to prevent exposing textual information to phonetic representations. Besides, we propose a novel self-distillation module for DORM to avoid overfitting pinyin information. Extensive experiments on three widely-used CSC datasets show that this model outperforms existing state-of-the-art baselines. Detailed analysis and studies show that direct interaction between characters and pinyin is beneficial to better restore misspelled characters. Through this work, we demonstrate the merit of disentangling phonetic features from textual representations when solving CSC.

Acknowledgements

We appreciate the anonymous reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China (No. 62176270), the Guangdong Basic and Applied Basic Research Foundation (No. 2023A1515012832), and the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (No. 2017ZT07X355).

Limitations

The potential limitations of our model are threefold. First, the training process requires more computational cost as the model needs to conduct two forward passes for each sample in the self-distillation module. Second, there is still room for improvement to reduce the model's overcorrection of legal characters. Third, the phonetics-aware sequence doubles the length of the original input, which demands extra computation cost at inference time.

Ethics Statement

This work aims to propose a technical method to utilize phonetic knowledge more effectively for Chinese Spelling Correction, which does not involve ethical issues. The datasets used in this work are all publicly available.

References

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. SpellGCN: Incorporating phonological and visual similarities into language models for Chinese spelling check. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 871–881, Online. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jianfeng Gao, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. 2010. A large scale ranker-based system for search query spelling correction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 358–366, Beijing, China. Coling 2010 Organizing Committee.

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, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

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.

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

Tuo Ji, Hang Yan, and Xipeng Qiu. 2021. SpellBERT: A lightweight pretrained model for Chinese spelling check. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3544–3551, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiahao Li, Quan Wang, Zhendong Mao, Junbo Guo, Yanyan Yang, and Yongdong Zhang. 2022a. Improving chinese spelling check by character pronunciation prediction: The effects of adaptivity and granularity. *arXiv preprint arXiv:2210.10996*.

Yinghui Li, Shirong Ma, Qingyu Zhou, Zhongli Li, Li Yangning, Shulin Huang, Ruiyang Liu, Chao Li, Yunbo Cao, and Haitao Zheng. 2022b. Learning from the dictionary: Heterogeneous knowledge guided fine-tuning for chinese spell checking. *arXiv preprint arXiv:2210.10320*.

Yinghui Li, Qingyu Zhou, Yangning Li, Zhongli Li, Ruiyang Liu, Rongyi Sun, Zizhen Wang, Chao Li, Yunbo Cao, and Hai-Tao Zheng. 2022c. The past mistake is the future wisdom: Error-driven contrastive probability optimization for Chinese spell checking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3202–3213, Dublin, Ireland. Association for Computational Linguistics.

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

Shulin Liu, Shengkang Song, Tianchi Yue, Tao Yang, Huihui Cai, TingHao Yu, and Shengli Sun. 2022. CRASpell: A contextual typo robust approach to improve Chinese spelling correction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3008–3018, Dublin, Ireland. Association for Computational Linguistics.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Bruno Martins and Mário J. Silva. 2004. Spelling correction for search engine queries. In *Advances in Natural Language Processing*, pages 372–383, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hossein Mobahi, Mehrdad Farajtabar, and Peter Bartlett. 2020. Self-distillation amplifies regularization in hilbert space. In *Advances in Neural Information Processing Systems*, volume 33, pages 3351–3361. Curran Associates, Inc.

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.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

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, Online. 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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 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, Online. 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.

Heng-Da Xu, Zhongli Li, Qingyu Zhou, Chao Li, Zizhen Wang, Yunbo Cao, Heyan 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, Online. 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.

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan 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, Online. Association for Computational Linguistics.

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

Xiaotian Zhang, Hang Yan, Sun Yu, and Xipeng Qiu. 2022. Sdcl: Self-distillation contrastive learning for chinese spell checking. *arXiv preprint arXiv:2210.17168*.

Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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, Dublin, Ireland. Association for Computational Linguistics.

A Pre-training

There are 1 million and 0.7 million articles in wiki2019zh corpus and weixin-public-corpus, respectively. First, we generate continuous sentence fragments of at most 256 characters from two corpora as pre-training samples. Then, we randomly sample 15% characters in each fragment and replace them with: (1) a phonologically similar character 80% of the time, (2) a randomly selected character 10% of the time, and (3) unchanged 10% of the time. After that, we acquire the pinyin sequence of the corrupted fragment and construct a phonetics-aware sequence, and replicate the original fragment to construct the prediction labels. We obtain a total of 4.8 million samples for pre-training.

The architecture of the model for pre-training is the same as described in Section 3.2. The model is trained by recovering those selected characters from the phonetics-aware sequence and the pinyin-to-character objective, while the self-distillation module is not required. The batch size is set to 72 and the learning rate is 5e-5.

B Datasets and Evaluation Metrics

The statistics of the training and test datasets for the experiments are presented in Table 6. It is worth mentioning that we post-process the predictions of characters “的”, “得” and “地” on the SIGHAN13 test set following previous work (Xu et al., 2021), because the annotations for these characters are not accurate. Specifically, the detection and correction of the three characters are not considered.

C Implementation of DROM

Our encoder contains 12 attention heads with a hidden size of 768 (about 110M parameters) and is initialized with weights from Chinese BERT-wwm (Cui et al., 2020). The embeddings of initials and finals are randomly initialized. Our model is firstly pre-trained and then fine-tuned on the CSC training set. We apply the AdamW optimizer (Loshchilov and Hutter, 2017) to fine-tune the model for 3 epochs on three 24G GeForce RTX 3090 GPUs. The learning rate is scheduled to decrease gradually after linearly increasing to 75e-6 during warmup. The maximum sentence length is set to 140. The batch sizes for training and evaluation are set to 48 and 32, respectively. The hyperparameters of α, β, and γ are set to 1, 1.2 and 0.97, respectively. Our implementation is based on Huggingface's Transformer (Wolf et al., 2020) in PyTorch.

Train	#Sent	#Errors	Avg. Length
SIGHAN15	2,338	3,037	31.3
SIGHAN14	3,437	5,122	49.6
SIGHAN13	700	343	41.8
271K pseudo data	271,329	381,962	42.6
Test	#Sent	#Errors	Avg. Length
SIGHAN15	1,100	703	30.6
SIGHAN14	1,062	771	50.0
SIGHAN13	1,000	1,224	74.3

Table 6: Statistics of the SIGHAN training and test datasets. We train our model on the combination of all the training sets and evaluate it on each test dataset.