ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240952

A Hybrid Approach Towards Chinese Spelling and Splitting Error Correction

Junhong Liang[1,2], **Junnan Zhu**[1,2,*], **Feifei Zhai**[1,3], **Nanchang Cheng**[4], **Chengqing Zong**[1,2] **and Yu Zhou**[1,3,*]

[1]State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, CAS, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3]Fanyu AI Laboratory, Zhongke Fanyu Technology Co., Ltd, Beijing, China
[4]National Broadcast Media Language Resources Monitoring and Research Center, Communication University of China, Beijing, China
liangjunhong2022@ia.ac.cn

Abstract. Existing Chinese spelling check (CSC) methods have limitations in correcting variable-length error characters, requiring the input and output to be the same length. They mainly focus on modelling Chinese characters' phonetic information and generating candidates for each position. In contrast, few approaches delve into the intricacies of splitting Chinese characters to address glyph errors and splitting variable-length corrections. We define the Chinese Splitting Error Correction (CSEC) task and develop CSEC datasets in news and social media domains to address this issue. We then propose **So**ft-Masked **Mu**lti-feature Error Correction (SoMu) model, which first generates semantic, phonetic, graphic, and unique Chinese Wubi embeddings, then integrates those features through selective gating fusion, followed by a soft-mask strategy to filter incorrect tokens and finally use transformer layers to predict the correct ones. This model effectively addresses both spelling and splitting errors. Extensive analysis shows that our model significantly improves character-splitting information modelling for CSEC. Our dataset is available at https://github.com/Skywalker-Harrison/SoMu.

1 Introduction

Spelling errors are prevalent daily and are usually caused by misidentification of Automatic Speech Recognition (ASR), Optical Character Recognition (OCR), or keyboard typing systems since some Chinese characters are similar in either phonology or morphology. Chinese Spelling Checking (CSC) aims to rectify the wrong Chinese characters in the sentence [25, 22].

In addition to spelling errors, splitting errors also appear in Chinese text recognition and editing since Chinese characters have unique compositions. For example, splitting errors occur in handwriting [28] when the spacing between the user's handwritten Chinese characters is relatively large, or the model over-segments an image. Splitting errors also occur in printed Chinese characters [30] and ancient text recognition [15] when dealing with low-quality images (e.g., being blurry, poorly lit, or noisy). Splitting also appears in online chatting; some people on the Internet may intentionally use split characters, such as typing “妈” as “女马” and “种” as “禾中”, which poses challenges for semantic understanding or other language processing tasks.

* Corresponding Author

Task	Source Sent.	Target Sent.
CSC	希望你们好好跳无(wú) (Phonetic error)	希望你们好好跳舞(wǔ) I hope you **dance** well.
CSC	这件商品兔费 (Graphic error)	这件商品免费 This item is **free**.
CSEC	请不要乱扔土立圾 (Splitting error)	请不要乱扔垃圾 Please do not **litter**!
CSEC	这些是直八正的创新 (Splitting error)	这些是真正的创新 These are **real** innovations.

Table 1: An example of CSC and CSEC Task, the original and target sentences are shown along with their English translation. The wrong character is shown in red color and the corrected character is shown in blue color. The error type is noted below the original sentence. The English translation of error token is highlighted in **bold**, best viewed in color.

Contrasting with traditional Chinese Spelling Check, which primarily corrects individual incorrect Chinese characters using phonetic or glyphic similarities, CSEC tackles errors related to character splitting. In other words, CSEC algorithmically explores merging multiple split characters into the correct one. While CSC typically maintains the same length between the input and the output, CSEC may result in a shorter output.

Table 1 demonstrates examples for CSC and CSEC. For the CSC, in the first example, the phonetic error “无”(wú, nothing) needs to be corrected as “舞” (wǔ, dance), and the second example requires the graphic error “兔” (rabbit) be corrected as “免” (free). Those two CSC examples only involve the correction of individual Chinese characters, and the correction pair (无-舞 and 兔-免) may share phonological or glyphic similarities. Table 1 also demonstrates two CSEC examples. In the first example, “土立” is separated from “垃” (rubbish). In the second example, “直八” is separated from “真” (real), where two split characters are combined to restore the original, correct character.

Current methods focus mainly on CSC, originating from the assumption that 1) the equivalence in length between input and output sequences [8], and 2) the prevalence of phonetic errors associated with Chinese characters, with comparatively limited occurrence of glyph-related errors [1]. However, those assumptions do not hold true in real-world applications, and the less common glyph-related errors adversely affect the user's satisfaction and relevant downstream tasks. Therefore, it is crucial to investigate further and analyze the glyph error category in CSC tasks to enhance the robustness and effectiveness of the existing techniques. Another challenge with existing models is that there is no separate multi-feature error detection module, so the model has to generate candidates for correction at each position, regardless of whether it is correct or not.

To address the aforementioned challenges, we propose a SoMu (**So**ft-Masked **Mu**lti-feature Error Correction) model to reliably detect and correct both spelling and splitting errors in Chinese text. Our contributions include:

- We define the CSEC task. To our knowledge, this is an important but untapped area in this field.
- We propose a novel SoMu model to address both CSC and CSEC tasks. It integrates semantic, phonological, and glyph features, which also includes Chinese Wubi segmentation of Chinese characters. Furthermore, it employs a soft-mask strategy to identify potential erroneous tokens and leverages transformer layers for making predictions.
- We develop the CSEC datasets and tailor CSC systems for CSEC tasks. In our experiments for CSC, the model is trained on SIGHAN training data and Wang271k data and tested on SIGHAN test data. Additionally, we use our own CSEC dataset for CSEC and thoroughly evaluate our model against other baseline models. The experimental results demonstrate that our model is comparable to the state-of-the-art model in CSC and excels in CSEC tasks.

2 Related Work

Spelling error correction aims to correct wrong tokens in the text. Early works such as Hanspeller [25] are mainly rule-based. Sequence Labeling [8] regards CSC as a sequence tagging problem and proposes the BiLSTM-CRF method to predict the best tag sequence. With the great success of pre-trained language models, BERT [7] has been used for CSC. FASPell [11] uses a BERT-based denoising autoencoder and a decoder to build up confusion set and combines phonological and graphical similarity to choose the correct character, Soft-Masked BERT [26] improves the detection ability of BERT model. GECTor [14] uses BERT-based models to predict the token transformation tag in English Error Correction (EEC). StructBERT [20] has been observed to produce satisfactory results in CSC [29].

In recent years, ReaLiSe [22] integrates semantic, phonetic, and visual modal selectively to deal with CSC. SCOPE [12] adds a character phonetic prediction module to better correct phonetic errors. PhVEC [10] proposes a non-autoregressive ASR correction method using pinyin tokens to extend the source sentence for variable-length correction. [9] adds acoustic information and uses MoE for modal alignment in EEC.

Although current models have achieved great success in the CSC task, previous studies have not sufficiently addressed glyph errors, character splitting, or employed multimodal soft masking techniques. We believe those features related to character splitting play a crucial role in addressing CSEC and improving the performance of CSC. Furthermore, the utilization of multimodal masking can enhance precision.

3 Methodology

3.1 Definitions

Existing Definition of CSC Given an input sentence $\boldsymbol{X} = [x_1, ..., x_n]$ consisting of n characters, the objective is to generate a corresponding output $\boldsymbol{Y} = [y_1, ..., y_n]$.

Definition of CSEC Given an input sentence $\boldsymbol{X} = [x_1, ..., x_m]$ comprising of m characters, there may exist one or more continuous sequence of characters, $x_i, ..., x_{i+l}$, to form a token y_i if merged together. Thus, the objective is to detect and correct these splitting errors, thereby restoring the original information in the output sentence $\boldsymbol{Y} = [y_1, ..., y_i, ..., y_n]$, where $n \leq m$.

3.2 The SoMu Model

As shown in Figure 1, our model can be divided into a feature fusion module, a detection module, and a correction module.

3.2.1 Feature Fusion Module

For the part of multi-feature integration and confusion, we follow the work in ReaLiSe [22], which incorporates semantic, graphic, and phonetic information, Chinese BERT model [5] is used to obtain the semantic embedding $\boldsymbol{E}^s$, a CNN encoder to obtain graphic embedding $\boldsymbol{E}^g$, a uni-directional GRU network [3] and transformer blocks to obtain phonetic embedding $\boldsymbol{E}^p$. The embedding for each feature is represented by a matrix of dimension $\mathbb{R}^{N\times D}$, where N represents the maximum length and D represents the hidden size.

Wubi Embedding The Wubi feature is incorporated into our system as it offers more efficient encoding of splitting information. The Wubi input method utilizes the QWERTY keyboard, with each key on the keyboard corresponding to different Chinese radicals[1]. Consequently, these codes represent sequential structural information, with a maximum length of 4 for each Chinese character. Moreover, the Wubi representation includes a Chinese splitting method that aligns with our CSEC. For instance, the Wubi code for the character “中” (middle) is “KH”, where “K” represents the radical “口” (mouth) and “H” represents the stroke “丨” (vertical bar) in Chinese. The PyWubi library[2] is employed for generating the Wubi codes of characters.

Then, suppose the Wubi code for the i-th character is $\boldsymbol{k}_i = (k_{i,1}, ..., k_{i,|\boldsymbol{k}_i|})$,where $|\boldsymbol{k}_i|$ is the length of Wubi code of i-th character, the Wubi embedding of the i-th character x_i is defined by

$$\tilde{h}^w_{i,j} = GRU(\tilde{h}^w_{i,j-1}, E(k_{i,j})) \tag{1}$$

where $\tilde{h}^w_{i,j}$ represents the hidden states of j-th position at character x_i, and $E(k_{i,j})$ is the embedding of Wubi code letter $k_{i,j}$, we use the last hidden state of the character, namely $\tilde{h}^w_{i,|\mathbf{k}_i|}$ as the character level Wubi representation of x_i.

Finally, the Wubi embedding for the input sentence is denoted as $\boldsymbol{E}^w \in \mathbb{R}^{N\times D}$

Feature Fusion The gating factor for semantic, graphic, phonetic and wubi features are denoted as g^s,g^g,g^p and g^w, and they will be calculated by the concatenation of embeddings.

We first start by calculating the overall representation of a sentence by averaging the semantic embedding in Eq. (2), then calculate selective gate weights in Eq. (3), and finally apply a selective gate feature fusion method to integrate semantic, phonetic, graphical, and Wubi

[1] https://en.wikipedia.org/wiki/Wubi_method
[2] https://pypi.org/project/pywubi/

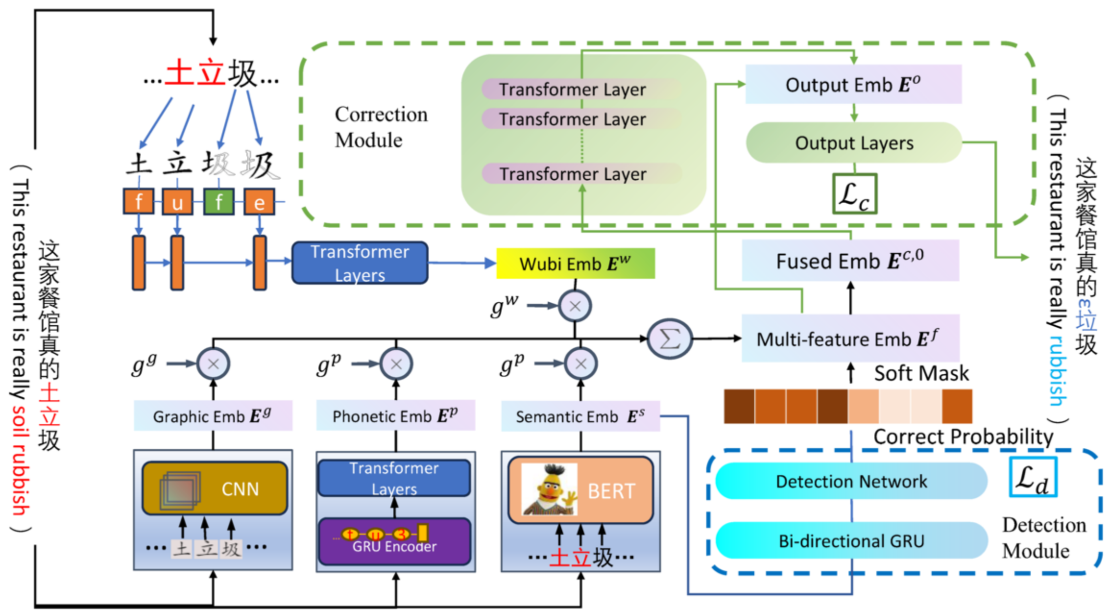

Figure 1: The architecture of our model, our model can be divided into three stages: (i) Selective feature fusion, (ii) Detection module (blue dashed border), and (iii) Correction module (green dashed border).

features in Eq. (4). The gate will control how much information from each feature will be fused into the final representation.

$$\bar{\boldsymbol{E}}^s = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{E}_i^s \tag{2}$$

$$g_i^{\{s,p,g,w\}} = \sigma(\boldsymbol{W}^{\{s,g,p,w\}}\cdot[\boldsymbol{E}_i^s,\boldsymbol{E}_i^g,\boldsymbol{E}_i^p,\boldsymbol{E}_i^w,\bar{\boldsymbol{E}}_i^s] + b^{\{s,p,g,w\}}) \tag{3}$$

$$\tilde{e}_i = g_i^s\cdot\boldsymbol{E}_i^s + g_i^g\cdot\boldsymbol{E}_i^g + g_i^p\cdot\boldsymbol{E}_i^p + g_i^w\cdot\boldsymbol{E}_i^w \tag{4}$$

where $\boldsymbol{W}^{\{s,g,p,w\}}\in\mathbb{R}^{N\times D}, b^{\{s,g,p,w\}}\in\mathbb{R}^N$ are all learnable parameters, s,p,g,w respectively denotes semantic, phonetics, graphical and wubi features. σ is the sigmoid function, and $[\cdot]$ denotes the concatenation of vectors. $\tilde{e}_i$ is the fused representation at i-th position. Then the fused vectors are packed into $\boldsymbol{E}^f = [\tilde{e}_1,...,\tilde{e}_N]$.

3.2.2 Detection and Correction Module

We use a soft masking strategy at the detection module to help our model focus on the wrong tokens. A bidirectional GRU network is firstly utilized to detect the correctness of tokens, using the semantic embedding $\boldsymbol{E}^s = \{e_1^s,...,e_N^s\}\in\mathbb{R}^{N\times D}$ as the input. The hidden states of GRU can be expressed as

$$h_i = [\overrightarrow{h_i},\overleftarrow{h_i}] = GRU(\boldsymbol{E}^s, e_i^s) \tag{5}$$

where i indicates the position and $\overrightarrow{h_i},\overleftarrow{h_i}$ incidates left to right and right to left hidden states.

Then, the GRU hidden states go through a feedforward neuron network to obtain the correct probabilities.

$$p_i^d = \sigma(\boldsymbol{W}_d h_i + b_d) \tag{6}$$

where $\boldsymbol{W}_d, b_d$ are learnable parameters. p_i^d denotes the correct probability of the i_{th} token.

Finally, we apply a soft-mask strategy to the gate-connected fused hidden embedding.

$$\boldsymbol{E}_i^{c,0} = p_i^d * \boldsymbol{E}_i^f \tag{7}$$

The detection loss is defined as:

$$\mathcal{L}_d = -\sum_{i=1}^{n}(p_i^d log p_i^d + (1-p_i^d)log(1-p_i^d)) \tag{8}$$

For the correction module, We start by feeding the embedding $\boldsymbol{E}^{c,0}$ from Eq. (7) to transformer blocks [18]

$$\boldsymbol{E}_i^{c,l} = Transformer_l(\boldsymbol{E}_i^{c,l-1}), l\in[1,L_c] \tag{9}$$

where l denotes the layer of the transformer network, and L_c is the number of transformer layers. $\boldsymbol{E}_i^{c,l}$ denotes the hidden state of i-th position at l_{th} layer.

Then, the residual connection is employed to add the original fused embedding $\mathbf{E}^f$ and transformer output $\mathbf{E}^{c,L_c}$

$$\boldsymbol{E}^o = \boldsymbol{E}^{c,L_c} + \boldsymbol{E}^f \tag{10}$$

We use the output hidden state to predict the token through a feed-forward neuron network to obtain the correction probability.

$$p_i^c = \sigma(\boldsymbol{W}_c\boldsymbol{E}_i^o + b_c) \tag{11}$$

where $p_i^c\in\mathbb{R}^{|V|}$, $|V|$ denotes the vocab size. $\boldsymbol{W}_c$ and b_c are learnable parameters of FFN. Then, we obtain the probability of i_{th} output token being corrected as j_{th} token in the vocabulary.

$$P(y_i = j|X) = p_i^c[j] \tag{12}$$

In this part, we define the correction loss by cross-entropy loss.

$$\mathcal{L}_c = -\sum_{i=1}^{n} logP(y_i = \hat{y}_i|X) \tag{13}$$

where $\hat{y}_i$ is the golden label at i_{th} position. In CSEC, we add placeholders to ensure the same output length (See 3.3). Then, we define the overall objective as follows:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_d + \alpha\mathcal{L}_c \tag{14}$$

where α is a hyper-parameter.

3.3 Modification on CSEC

To address the errors related to the splitting of Chinese tokens in CSEC, certain modifications have been made: 1) we exclude phonetic and graphic features because they do not indicate splitting information explicitly. 2) we adjust the model's output by placing the original character at the last position among the split characters while using a placeholder symbol ε for the other positions. For instance, the sentence "这家餐馆真的土立圾" is corrected as "这家餐馆真的ε垃圾". Subsequently, a post-processing module removes the ε placeholders. All other settings remain unchanged.

3.4 Pretraining Wubi Encoder

To leverage the Wubi encoder, we devise a sentence transformation task to predict the corresponding Chinese characters using the Wubi encoding of a sentence. The Wubi encoder is the same as illustrated in Figure 1. This task emulates the human typing process, allowing the Wubi encoder to inherently capture information regarding character splitting. During the pre-training phase, a linear layer is implemented on the encoder to predict the output of Chinese characters.

For CSEC tasks, we add a weighting parameter w when calculating detection loss $\mathcal{L}_d$

$$\mathcal{L}_d = -\sum_{i=1}^{n} \left(p_i^d \log p_i^d + w \cdot (1 - p_i^d) \log(1 - p_i^d) \right) \tag{15}$$

4 Experiment

4.1 Dataset

CSC Data We collect data from SIGHAN13 [21], SIGHAN14 [24], and SIGHAN15 [16] dataset and the pseudo training data created by ASR and OCR confusion method from Wang271K [19] using confusion sets. We conduct joint training on the SIGHAN13/14/15 train set and Wang271K and evaluate the model on the SIGHAN14/15 test set, while the SIGHAN13 test set is excluded since its poor annotation in Chinese modifiers such as "的地得".

CSEC Data We curate two datasets, namely CSEC-Social and CSEC-News, to investigate the splitting of Chinese characters in social media and standard texts. CSEC-Social is created utilizing malicious comments on COLDataset [6] collected from the Chinese social network Weibo. CSEC-News is collected from the Sougou News dataset [27]. Finally, we employ character splitting dictionary [3] to obtain the split result. This segmentation process is conditional; a character is identified if all of its splits are recognized by the vocabulary of the pre-trained chinese-bert-wwm model [4]. Each identified character is subjected to splitting with a probability of 0.1.

Table 2 provides an exhaustive compilation of statistical information about the CSC and CSEC datasets.

4.2 Metrics

CSC involves two primary levels of metrics: character level and sentence level. In the case of sentence-level assessment, a sentence is deemed correct only if all incorrect characters within it are successfully corrected. Following previous approaches, we adopted the sentence-level metrics presented in [22].

[3] https://github.com/kfcd/chaizi

Task		Name	#sent	Avg Len	#Errors
CSC	Train	SIGHAN14	3,437	49.6	5,122
		SIGHAN15	2,338	31.3	3,037
		Wang271K	271,329	42.6	381,692
		Total	277,804	42.6	390,464
	Test	SIGHAN14	1,062	50.0	771
		SIGHAN15	1,100	30.6	703
		Total	2,162	40.1	1,474
CSEC	Train	News	7,869	30.7	8,344
		Social	6,330	28.8	6,401
		Total	14,199	29.9	14,745
	Dev	News	775	29.3	800
		Social	1,669	28.9	1,728
		Total	2,444	29.0	2,528
	Test	News	1,174	30.9	1,268
		Social	1,108	28.6	1,128
		Total	2,282	29.8	2,396

Table 2: Statistics of the CSC and CSEC datasets.

For CSEC, most of the input and output sentences are not of equal length because they involve the combination of split tokens. Therefore, to address variable-length evaluation, we use ChERRANT [29] as our evaluation criterion. ChERRANT extracts the editing distance between original and corrected sentences and calculates the Levenshtein distance of the predicted edits.

4.3 Baselines

We compare SoMu with the following baselines for CSC.

- **HanSpeller++** [25] uses traditional statistical methods, including candidate generation, re-ranking and global selection, to correct the error.
- **BERT** [7, 26] fine-tunes the BERT language model using CSC dataset.
- **FASPell** [11] uses a denoising autoencoder and a decoder to generate predicted tokens.
- **SpellGCN** [2] incorporates phonological and visual similarity knowledge using a graph convolutional network.
- **SM BERT** [26] incorporates a soft-masked BERT model for the detection and correction of error characters.
- **ReaLiSe** [22] constructs a multimodal method and mixes semantic, graphical, and phonetical information.
- **ECOPO** [13] adds a plug-in module to guide the model to avoid predicting common characters in an error-driven way.
- **SCOPE** [12] adaptively integrates both CSC and Chinese Phonetic Predicting tasks, focusing on phonetic features.
- Multidimensional Multimodal Integration (**MMI**) [23] which combines phonetic and structural forms of Chinese characters, then integrates with semantic information[4].

Since both SCOPE and ECOPO are plug-in modules, we record their performance on the baseline Realize.

For CSEC, the baselines are listed as follows:

- **Seq2Seq** [29] regards error correction as an autoregressive unidirectional language generation.
- **GECTor** [14] regards the correction problem as label tagging.

[4] In the article, the authors do not indicate the name of their model, here we use MMI to denote the model proposed by [23]

Meanwhile, we also use **ReaLiSe** [22] and **SCOPE** [9] for comparison.

In our research, we also opt for Large Language Models (LLMs) for both CSC and CSEC tasks. We specifically choose OpenAI ChatGPT and other open-sourced language models. The task prompts can be found in Appendix A. We test those models on a 5-shot scenario.

- **GPT-3.5, GPT-4 and GPT-4o** from the ChatGPT series. We have selected models accessible through OpenAI's public APIs[5][6]. GPT-3.5 model benefits from extensive training on a broad dataset using GPT-3 with Reinforcement Learning from Human Feedback. While GPT-4 has more training data and model parameters, and GPT-4o demonstrated better ability on text, audio and vision.
- **Baichuan2-13B-Chat** Given our focus on Chinese scenarios, we also incorporate LLMs that are prominent and extensively utilized within the Chinese community, we include Baichuan2-13B-Chat[7] in our experiment.
- **Qwen2** Qwen2 is the new model series of large language models from the Qwen team. We include Qwen2-7B-Instruct [8] in our experiment.

4.4 *Implementation Details*

We use Pytorch to implement the SoMu model. We initialize the semantic encoder using Chinese-BERT-wwm [4] and further pre-train the graphic encoder, phonetic encoder, and Wubi encoder. In the correction module, we set $L_c = 6$. For CSC, our model is trained in two stages: in the first stage, we train our model on SIGHAN/Wang271k joint training data for ten epochs, we choose the batch size to be 16, use AdamW optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$), and set the learning rate to 5×10^{-5}. In the second stage, we fine-tune our model using only SIGHAN training data, and the learning rate is set to 6×10^{-5}. For CSEC, we only conduct the first stage training using the CSEC dataset and set $w = 10$. All experiments are performed on two GeForce RTX 3090s.

4.5 *Main Results*

CSC Table 3 and 4 show the performance of our proposed model together with other baselines. Our model demonstrates superior performance compared to the ReaLiSe baseline on the SIGHAN14/15 benchmarks, achieving a 1% improvement in the F_1 score. While our model does not surpass ECOPO and SCOPE in the SIGHAN14 benchmark, it is noteworthy that ECOPO is model-agnostic, suggesting that a combination with our model could potentially enhance performance. For LLMs, despite the constraints on the format of output, the performance still falls far short of the small models on the evaluation metrics; demonstrating CSC is still a challenging task for LLMs. Our model's high precision score on SIGHAN14 and its accuracy score on SIGHAN15 are attributable to the efficacy of our soft-mask module in discerning character correctness.

CSEC Table 5 shows the performance of our model and all baselines using ChERRANT. Our model excels in the reconstruction of split characters within both the CSEC-Social and CSEC-News datasets. While Seq2Seq has shown promise in Grammatical Error Correction (GEC) tasks, it performs poorly in error correction with hard constraints. GECTor, in particular, underperforms due to its inability to identify incorrect tokens accurately. In contrast, once our model successfully identifies an error, it demonstrates a high correction probability of 95% on both datasets. Additionally, we note that despite SCOPE's leading position in CSC, it lags significantly behind our SoMu model in CSEC. This shortfall is attributed to SCOPE's lack of explicit glyph integration in its framework.

We observed that LLMs including *gpt-3.5-turbo* and *Baichuan2-13B-chat* often fall short of expectations because they prioritize semantically correct modifications to sentences instead of accurately restoring original split characters, making unnecessary edits. Additionally, these models struggle to effectively incorporate glyph information into their processing, leading to an unsatisfactory perfor-

SIGHAN14	Detection Level				Correction Level			
	Acc	Pre	Rec	F_1	Acc	Pre	Rec	F_1
FASpell	70.0	61.0	53.5	57.0	69.3	59.4	52.0	55.4
BERT	73.8	60.2	64.6	62.3	73.0	58.6	62.9	60.7
SpellGCN	-	65.1	69.5	67.2	-	63.1	67.2	65.3
ECOPO	79.0	68.8	72.1	70.4	71.5	67.5	71.0	69.2
SCOPE	-	69.0	75.0	71.9	-	67.1	72.9	67.9
MMI	-	67.1	66.7	66.9	-	66.3	66.0	66.2
GPT3.5	-	25.7	30.0	27.7	-	19.8	11.2	14.3
GPT4	-	-	-	-	-	24.1	15.9	19.1
GPT4o	-	-	-	-	-	38.4	22.4	28.3
Baichuan	-	10.1	16.7	12.6	-	8.94	14.8	11.2
Qwen2	-	-	-	-	-	10.4	5.60	7.20
ReaLiSe	78.4	67.8	71.5	69.6	77.7	66.3	70.0	68.1
SoMu (Ours)	78.7	**68.8**	71.6	70.3	**78.3**	**67.8**	70.2	69.0

Table 3: The sentence level performance of our model and other baselines on SIGHAN14 dataset. Best results are in **bold**. Underlined results indicate the model's performance exceeds the ReaLiSe baseline.

SIGHAN15	Detection Level				Correction Level			
	Acc	Pre	Rec	F_1	Acc	Pre	Rec	F_1
HanSpeller++	70.1	80.3	53.3	64.0	69.2	79.7	51.5	62.5
FASpell	74.2	67.6	60.0	63.5	73.7	66.6	59.1	62.6
SM BERT	80.9	73.7	73.2	73.5	77.4	66.7	66.2	66.4
BERT	82.2	72.8	77.8	75.3	81.5	71.5	76.3	73.8
SpellGCN	-	74.8	80.7	77.7	-	72.1	77.7	75.9
ECOPO	85.0	77.5	82.6	80.0	84.2	76.1	81.2	78.5
SCOPE	-	78.7	84.7	81.6	-	76.8	82.6	79.6
MMI	-	76.7	77.1	76.9	-	76.1	76.5	76.3
GPT3.5	-	-	-	-	-	31.7	19.8	23.5
GPT4	-	-	-	-	-	40.4	25.9	31.6
GPT4o	-	-	-	-	-	50.2	32.8	40.3
Baichuan	-	19.5	33.5	24.6	-	18.6	32.0	23.6
Qwen2	-	-	-	-	-	15.4	8.90	11.3
ReaLiSe	84.7	77.3	81.3	79.3	84.0	75.9	79.9	77.8
SoMu (Ours)	**85.7**	78.3	82.9	80.5	**84.7**	76.5	81.0	78.9

Table 4: The sentence level performance of our model and other baselines on SIGHAN15 dataset. Best results are in **bold**. Underlined results indicate the model's performance exceeds the ReaLiSe baseline.

	CSEC-Social			CSEC-News		
Model	Pre	Rec	$F_{0.5}$	Pre	Rec	$F_{0.5}$
Seq2Seq	15.9	22.8	16.9	27.1	30.5	27.7
GECTor	48.7	77.1	52.6	69.8	82.5	72.0
BERT	55.6	92.1	60.4	90.2	94.5	91.0
SCOPE	56.3	89.6	60.8	91.1	91.6	91.2
GPT3.5	8.30	12.5	8.90	21.2	24.7	21.8
GPT4	18.4	7.80	14.5	54.6	56.3	54.9
GPT4o	35.5	43.1	36.8	56.2	57.5	56.4
Baichuan	13.6	24.9	15.0	27.2	36.5	28.7
ReaLiSe	59.5	94.4	64.2	94.3	93.9	94.2
SoMu (Ours)	**60.2**	**95.1**	**65.0**	**96.0**	**96.1**	**96.0**

Table 5: The performance of our model and all baseline models on CSEC test sets. Best results are in **bold**.

[5] https://chat.openai.com/

[6] We used OpenAI APIs, the model for GPT-3.5 is gpt-3.5-turbo-0401, for GPT-4 is gpt-4-turbo-2024-04-09 and for GPT-4o is gpt-4o.

[7] https://github.com/Baichuan2-inc/Baichuan2

[8] https://huggingface.co/Qwen/Qwen2-7B-Instruct

mance in correcting the true errors.

We also find that the model performs significantly better in precision and $F_{0.5}$ score on CSEC-Social compared to CSEC-News because CSEC-Social exhibits a higher prevalence of non-standard expressions compared to CSEC-News, resulting in lower precision for the model in identifying splitting characters.

4.6 Ablation Study

CSC The results of ablation experiments for CSC are shown in Table 6. We verify the effectiveness of each part of our model by conducting the ablation study with the following settings: 1) remove the phonetic encoder, 2) remove the graphic encoder, 3) remove the Wubi encoder, and 4) remove the soft-masking strategy.

As shown in Table 6, the removal of phonetic, graphic, Wubi, and soft mask would decrease the performance, which demonstrates the effectiveness of our model. This study particularly highlighted that adding Wubi embedding and soft-mask strategy and removing the Wubi encoder would lead to a two-level drop in the performance in detection and correction level, which is a more substantial impact than the removal of phonetic and graphical information.

Detection Level				
Model	Acc	Pre	Rec	F1
BERT	78.4	74.6	74.5	74.5
SoMu	85.7	78.3	82.9	80.5
- w/o Phonetic	84.7	76.9	81.7	79.2
- w/o Graphic	85.1	78.5	81.5	80.0
- w/o Wubi	83.9	76.6	80.0	78.3
- w/o Soft Masking	84.9	77.4	81.7	79.5
Correction Level				
BERT	76.9	72.3	72.3	72.3
SoMu	84.7	76.5	81.0	78.9
- w/o Phonetic	83.7	75.0	80.0	77.2
- w/o Graphic	84.3	76.9	79.9	78.3
- w/o Wubi	83.0	74.9	78.2	76.5
- w/o Soft Masking	84.0	75.7	79.9	77.7

Table 6: Ablation experiments of SoMu on SIGHAN15 test data with different settings, we report the detection and correction level score. The underlined number represents the configuration with the greatest reduction in performance.

CSEC We also conduct the ablation study on CSEC test datasets. The results are shown in Table 7. The SoMu model is modified according to section 3.3. The removal of Wubi information will lead to a decrease in overall performance. We also find that adding phonetic features leads to the most drop-in $F_{0.5}$ score, and adding graphic and phonetic information leads to a drop in the overall performance. A possible reason is that phonetic information is unnecessary in CSEC tasks, adding it may lead to sub-optimal performance.

5 Analysis

5.1 Character Embedding

To understand why SoMu generates meaningful representations, we study the fused hidden embedding $\mathbf{E}^f$ of each character. We select characters with similar shapes and display two separate t-SNE visualizations [17]. Figure 2 compares the clustering of data points using two different models: BERT on the left and SoMu on the right. SoMu can distinguish the same shape characters much better than the BERT model. Take the clustering of “于” (at) and “干” (do) as an example. The t-SNE visualization of SoMu forms more distinctive clusters than BERT, and the characters with similar shapes form two clusters.

CSCD-Social			
Model	Pre	Rec	$F_{0.5}$
ReaLiSe	59.5	94.4	64.2
SoMu *	60.2	95.1	65.0
- w/o Wubi	59.8	94.4	64.5
- w Graphic	59.7	94.2	64.4
- w Phonetic	59.6	94.4	64.3
CSCD-News			
Model	Pre	Rec	$F_{0.5}$
ReaLiSe	94.3	93.9	94.2
SoMu *	96.0	96.1	96.0
- w/o Wubi	95.6	95.4	95.5
- w Graphic	95.6	95.3	95.5
- w Phonetic	94.8	94.1	94.7

Table 7: Ablation experiments of SoMu on CSEC test data with different settings, we report the detection and correction level score. The underlined number represents the configuration with the greatest reduction in performance. We denote our modified SoMu model without phonetic and graphic embedding in *.

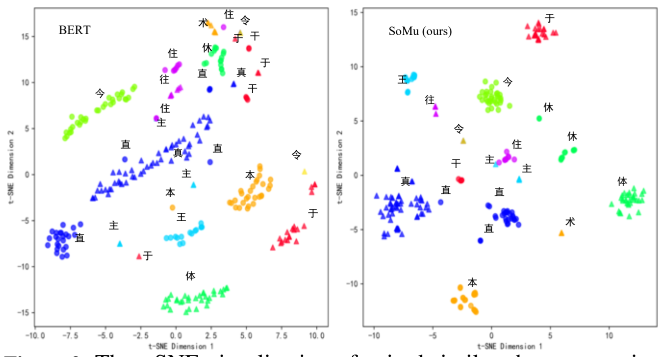

Figure 2: The t-SNE visualization of paired similar characters using BERT (left) and our SoMu (right) in the SIGHAN15 test. The paired similar characters are marked in the same color but in circle ○ and triangle △. Best viewed in color.

5.2 Parameter tuning

We try four α values in $[0.2, 0.5, 0.8, 1]$ to balance detection and correction losses in Eq. (14), then report the character level correction and detection F_1 score before fine-tuning. The result is shown in Figure 3. Then, α is selected as 0.5 for our experiment based on the overall performance.

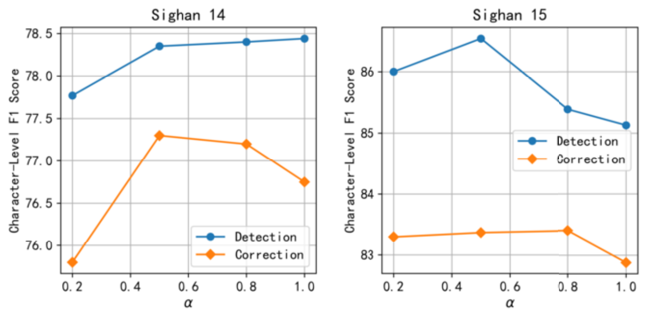

Figure 3: The Character level performance with different α values.

5.3 Multi-feature integration

We categorize three types of error: similar phonetics (SP), similar shape (SS), similar shape and phonetics (SSP) by automatically calculating the character similarity using Levenstein distance provided by FASPell [11]. We use MacBERT[9] for comparison. The results are shown in Table 8. From the table, SoMu outperforms MacBERT in correcting all three types of tokens. The reason behind this is multi-feature integration, which allows more information related to phonetics and glyphs to be fused into the overall representation of Chinese characters. We also analyze the capability of our model to correct splitting errors, and the result is shown in Table 9. From the table, our model excels in correcting splitting errors in both news and social media domains.

	SIGHAN14			SIGHAN15		
	SP	SS	SSP	SP	SS	SSP
Total	303	20	275	338	22	200
MacBERT	206	20	210	262	19	169
SoMu (Ours)	**222**	**20**	**218**	**277**	**21**	**186**

Table 8: Wrong character correction abilities comparison between MacBERT and SoMu on SIGHAN14/15 datasets. "Total" indicates the number of error tokens, and the number of corrected tokens of MacBERT and SoMu is listed. The **bold** numbers indicates the method that could correct the most wrong characters.

Dataset	News	Social	All
Total	1,268	1,128	2,396
SCOPE [9]	1,183	1,062	2,245
Realise [22]	1,220	1,094	2,314
SoMu (Ours)	**1,225**	**1,108**	**2,333**

Table 9: Splitting error correction abilities comparison among SCOPE, Realise, and SoMu on CSCD datasets. "Total" indicates the number of split tokens, and the numbers of corrected tokens of different methods are listed. The **bold** numbers indicate the method that could correct the most splitting characters.

5.4 Case Study

CSEC cases are shown in Table 10. SoMu could detect the wrong splits and recover the sentence by noticing the structure of the splits. Compared to ReaLiSe, SoMu correctly recovers "车甫" into "辅" (tutor) and "毛炎" into "毯" (carpet). Moreover, SoMu maintains the correct sentence in the third example, while ReaLiSe mistakenly merges "丰子" into "李". SoMu could avoid such error since the Wubi codes for "丰子" and "李" are significantly different. While GPT-3.5 fails to construct some characters since it cannot effectively embed the information of splitting parts of Chinese characters.

6 Conclusion

In this article, we introduce a hybrid model called SoMu, which combines multi-feature embedding and a soft masking strategy to simultaneously address the challenges of CSC and CSEC. Our model incorporates Chinese Wubi sequential information alongside other features, then identifies problematic characters with soft masking and ultimately makes predictions using a transformer network. The incorporation of the Wubi feature enables our model to concentrate on the segmentation of Chinese characters, and soft masking allows the model to focus on the wrong tokens. Experimental results demonstrate that our approach performs competitively with the state-of-the-art model in CSC and surpasses it in CSEC.

Example 1:	家长拒绝车甫导孩子或被计入征信。 Parents who refuse to tutor their children may be recorded for credit.
ReaLiSe:	家长拒绝捕导孩子或被计入征信。
GPT3.5:	家长拒绝车辆导孩子或被计入征信。
SoMu:	家长拒绝辅导孩子或被计入征信。
Example 2:	周一围木示志性的马尾和帅气大长腿，在红毛炎上十分瞩目。 Zhou Yiwei's iconic ponytail and handsome long legs attracted much attention on the red carpet.
ReaLiSe:	周一围标志性的马尾和帅气大长腿，在红毛炎上十分瞩目。
GPT3.5:	周一围木示志性的马尾和帅气大长腿，在红毯上十分瞩目。
SoMu:	周一围标志性的马尾和帅气大长腿，在红毯上十分瞩目。
Example 3:	丰子恺曾指出，小孩子的生活是趣味本位的。 Feng Zikai once pointed out that children' s life is based on fun.
ReaLiSe:	李恺曾指出，小孩子的生活是趣味本位的。
GPT3.5:	丰子恺曾指出，小孩子的生活是趣味本位的。
SoMu:	丰子恺曾指出，小孩子的生活是趣味本位的。

Table 10: Examples CSEC task for ReaLiSe and SoMu, we mark the split tokens/wrong edits/correct edits in red/orange/blue. Best viewed in color.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 62106263, Grant 62206286, and the Open Research Fund of National Broadcast Media Language Resources Monitoring and Research Center, Communication University of China. The authors would like to thank Dr. Xiaomian Kang for his insightful discussions.

A Task Prompts for LLMs

The prompts for generating knowledge keywords as web search queries were illustrated in Table 11.

CSC	你是一位著名的语言学家，请你纠正一下句子中汉字拼写的错误，你需要识别并纠正用户输的句中可能的错别字并输出正确的句子，纠正时必须保证改动前后句必须等长，在纠正错别字的同时尽可能减少对原句的改动（不添加额外标点符号，不添加额外的字，不删除多余的字）。只输出没有错别字的句，不要添加任何其他解释或说明。如果句没有错别字，就直接输出和输相同的句。下面，请纠正:
CSEC	你是一位著名的语言学家，请你纠正一下句子中汉字拆分错误，你需要识别并纠正用户输的句中可能的拆分字并输出正确的句子，纠正时必须保证被拆开的汉字得到还原，在纠正拆分字的同时尽可能减少对原句的改动（不添加额外标点符号，不添加额外的字，不删除多余的字）。只输出没有拆分字的句，不要添加任何其他解释或说明。如果句没有拆分字，就直接输出和输相同的句。 下面，请纠正:

Table 11: Prompts of CSC and CSEC tasks for LLMs

[9] https://huggingface.co/shibing624/macbert4csc-base-chinese

References

[1] H. Chen and X. Wang. Pygc: A pinyin language model guided correction model for chinese spell checking. In *International Conference on Neural Information Processing*, pages 224–239. Springer, 2023.

[2] X. Cheng, W. Xu, K. Chen, S. Jiang, F. Wang, T. Wang, W. Chu, and Y. Qi. 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. Association for Computational Linguistics, 2020.

[3] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[4] Y. Cui, W. Che, T. Liu, B. Qin, S. Wang, and G. Hu. 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, Nov. 2020. Association for Computational Linguistics.

[5] Y. Cui, W. Che, T. Liu, B. Qin, and Z. Yang. Pre-training with whole word masking for chinese bert. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 29:3504–3514, nov 2021. ISSN 2329-9290.

[6] J. Deng, J. Zhou, H. Sun, C. Zheng, F. Mi, H. Meng, and M. Huang. COLD: A benchmark for Chinese offensive language detection. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11580–11599, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[8] J. Duan, B. Wang, Z. Tan, X. Wei, and H. Wang. Chinese spelling check via bidirectional lstm-crf. In *2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, pages 1333–1336, 2019.

[9] T. Fang, J. Hu, D. F. Wong, X. Wan, L. S. Chao, and T.-H. Chang. Improving grammatical error correction with multimodal feature integration. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9328–9344, Toronto, Canada, July 2023. Association for Computational Linguistics.

[10] Z. Fang, R. Zhang, Z. He, H. Wu, and Y. Cao. Non-autoregressive Chinese ASR error correction with phonological training. In M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5907–5917, Seattle, United States, July 2022. Association for Computational Linguistics.

[11] Y. Hong, X. Yu, N. He, N. Liu, and J. Liu. 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. Association for Computational Linguistics, 2019.

[12] J. Li, Q. Wang, Z. Mao, J. Guo, Y. Yang, and Y. Zhang. Improving Chinese spelling check by character pronunciation prediction: The effects of adaptivity and granularity. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4275–4286, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

[13] Y. Li, Q. Zhou, Y. Li, Z. Li, R. Liu, R. Sun, Z. Wang, C. Li, Y. Cao, and H.-T. Zheng. The past mistake is the future wisdom: Error-driven contrastive probability optimization for Chinese spell checking. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3202–3213, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[14] K. Omelianchuk, V. Atrasevych, A. Chernodub, and O. Skurzhanskyi. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170. Association for Computational Linguistics, 2020.

[15] C.-W. Tang, C.-L. Liu, and P.-S. Chiu. Hrcenternet: An anchorless approach to chinese character segmentation in historical documents. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2020.

[16] Y.-H. Tseng, L.-H. Lee, L.-P. Chang, and H.-H. Chen. Introduction to SIGHAN 2015 bake-off for Chinese spelling check. In L.-C. Yu, Z. Sui, Y. Zhang, and V. Ng, editors, *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 32–37, Beijing, China, July 2015. Association for Computational Linguistics.

[17] L. van der Maaten and G. E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[19] D. Wang, Y. Song, J. Li, J. Han, and H. Zhang. A hybrid approach to automatic corpus generation for Chinese spelling check. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2517–2527, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics.

[20] W. Wang, B. Bi, M. Yan, C. Wu, Z. Bao, J. Xia, L. Peng, and L. Si. Structbert: Incorporating language structures into pre-training for deep language understanding. *arXiv preprint arXiv:1908.04577*, 2019.

[21] S.-H. Wu, C.-L. Liu, and L.-H. Lee. Chinese spelling check evaluation at SIGHAN bake-off 2013. In L.-C. Yu, Y.-H. Tseng, J. Zhu, and F. Ren, editors, *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42, Nagoya, Japan, Oct. 2013. Asian Federation of Natural Language Processing.

[22] H.-D. Xu, Z. Li, Q. Zhou, C. Li, Z. Wang, Y. Cao, H. Huang, and X.-L. Mao. Read, listen, and see: Leveraging multimodal information helps Chinese spell checking. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 716–728, Online, Aug. 2021. Association for Computational Linguistics.

[23] L. Yang, X. Liu, T. Liao, Z. Liu, M. Wang, X. Fang, and E. Yang. Is Chinese Spelling Check ready? Understanding the correction behavior in real-world scenarios. *AI Open*, 4:183–192, 2023. ISSN 2666-6510.

[24] L.-C. Yu, L.-H. Lee, Y.-H. Tseng, and H.-H. Chen. Overview of SIGHAN 2014 bake-off for Chinese spelling check. In L. Sun, C. Zong, M. Zhang, and G.-A. Levow, editors, *Proceedings of the Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 126–132, Wuhan, China, Oct. 2014. Association for Computational Linguistics.

[25] S. Zhang, J. Xiong, J. Hou, Q. Zhang, and X. Cheng. HANSpeller++: A unified framework for Chinese spelling correction. In L.-C. Yu, Z. Sui, Y. Zhang, and V. Ng, editors, *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 38–45, Beijing, China, July 2015. Association for Computational Linguistics.

[26] S. Zhang, H. Huang, J. Liu, and H. Li. Spelling error correction with soft-masked BERT. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 882–890, Online, July 2020. Association for Computational Linguistics.

[27] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 649–657, Cambridge, MA, USA, 2015. MIT Press.

[28] X.-Y. Zhang, Y.-C. Wu, F. Yin, and C.-L. Liu. *Deep Learning Based Handwritten Chinese Character and Text Recognition*, pages 57–88. 02 2019. ISBN 978-3-030-06072-5.

[29] Y. Zhang, Z. Li, Z. Bao, J. Li, B. Zhang, C. Li, F. Huang, and M. Zhang. MuCGEC: a multi-reference multi-source evaluation dataset for chinese grammatical error correction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3118–3130. Association for Computational Linguistics, 2022.

[30] H. Zheng, J. Wang, Z. Huang, Y. Yang, and R. Pan. Chinese/english mixed character segmentation as semantic segmentation. *arXiv preprint arXiv:1611.01982*, 2016.