ARM: An Alignment-and-Replacement Module for Chinese Spelling Check Based on LLMs

Changchun Liu[1], **Kai Zhang**[1*], **Junzhe Jiang**[1], **Zirui Liu**[1],
Hanqing Tao[2], **Min Gao**[3], **Enhong Chen**[1]
[1]State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
[2]School of Information and Control Engineering, China University of Mining
[3]The First Affiliated Hospital of University of Science and Technology of China
{changchun_liu, jzjiang, liuzirui}@mail.ustc.edu.cn
{kkzhang08, cheneh}@ustc.edu.cn, hqtao@cumt.edu.cn, gmbeauty@163.com

Abstract

Chinese Spelling Check (CSC) aims to identify and correct spelling errors in Chinese texts, where enhanced semantic understanding of a sentence can significantly improve correction accuracy. Recently, Large Language Models (LLMs) have demonstrated exceptional mastery of world knowledge and semantic understanding, rendering them more robust against spelling errors. However, the application of LLMs in CSC is a double-edged sword, as they tend to unnecessarily alter sentence length and modify rare but correctly used phrases. In this paper, by leveraging the capabilities of LLMs while mitigating their limitations, we propose a novel plug-and-play ***A**lignment-and-**R**eplacement **M**odule* (**ARM**) that enhances the performance of existing CSC models and without the need for retraining or fine-tuning. Experiment results and analysis on three benchmark datasets demonstrate the effectiveness and competitiveness of the proposed module.

1 Introduction

Chinese Spelling Check (CSC) is a fundamental Natural Language Processing (NLP) task behind many downstream applications, including web search (Gao et al., 2010; Martins and Silva, 2004). It aims to detect and correct spelling errors in Chinese texts, with a specific focus on alignment errors (Wu et al., 2013a). Alignment errors do not alter the length of the text, as corrections are made exclusively through the substitution of characters without the operation of addition or deletion. Typically, these errors originate from automatic speech recognition (ASR) or optical character recognition (OCR) systems, often involving the incorrect use of characters that are phonologically or visually similar (Liu et al., 2010).

According to the characteristics CSC errors, previous studies have primarily utilized non-autoregressive pre-trained language models (PLMs) and enhance PLMs by formulating custom-designed pre-training objectives (Zhang et al., 2021b; Li et al., 2022d; Liang et al., 2023; Liu et al., 2024b) or developing various methods to extract and integrate the phonetic and visual features of characters (Liu et al., 2021; Xu et al., 2021; Li et al., 2022c; Wei et al., 2023). Those studies typically feature models with relatively few parameters, which limits their ability to comprehend wrong or complex expressions. Additionally, the rigidity of their training processes restricts them to memorizing only a limited set of predefined modifications.

Figure 1: Examples of shortcomings of employing LLMs on Chinese Spelling Check. Incorrect characters are highlighted in red, with their correct counterparts provided in parentheses. Additionally, yellow indicates LLM-made modifications.

Recent advancements in Large Language Models (LLMs), such as GPT-4 (Achiam et al., 2023) and Gemini (Team et al., 2023), have garnered significant attention. Numerous evaluations (Chang et al., 2024; Liu et al., 2024a) demonstrate that

*Corresponding authors: Kai Zhang.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 10156–10168
November 12-16, 2024 ©2024 Association for Computational Linguistics

LLMs possess strong semantic understanding capabilities. Li et al. (2023) points out that LLMs have better domain adaptability and data tolerance ability than traditional CSC models, which means that LLMs have the capacity for context-sensitive adaptations rather than merely relying on rote memorization. However, the application of LLMs in CSC remains relatively unexplored.

The reason is that LLMs exhibits several key limitations, which leads to its poor performance on CSC as evidenced by the test results presented in Appendix A. Firstly, as autoregressive generative models, LLMs inherently generate outputs of variable lengths, which implies that LLMs may modify sentences through addition or deletion operations. Secondly, the outputs of LLMs are not always consistent, with a considerable likelihood that the responses may not conform to the required output format. Thirdly, due to their training method, LLMs tend to normalize correct but less common expressions into more frequently used equivalents, resulting in over-correction.

Figure 1 provides three examples of the limitations discussed above. In the first example, the character "跑 (run)" was erroneously substituted with "太熟了 (so deeply)" instead of the visually and phonetically similar "饱 (fully)". While the modified phrase retained the similar meaning, it altered the sentence length. In the second sentence, LLMs corrected the wrong character, but inappropriately prefixed the sentence with "修改后的句子是 (revised sentence)", which contradicts the CSC output specifications. In the third instance, despite being error-free, LLMs unnecessarily revise "服务生 (server)" to the more frequently used homonym "服务员 (waiter)", leading to over-correction.

To this end, we propose an ***A****lignment-and-****R****eplacement* ***M****odule* (**ARM**) based on LLMs for CSC, to enhance the performance of existing CSC models and resolve LLMs shortcomings in CSC. The proposed module is designed to be compatible with existing CSC models, without the need for retraining or fine-tuning. Specifically, to address the first and second shortcomings, we propose an alignment method (**ERS**) to align LLMs outputs, which is based on ***E****dit distance*, ***R****ecursion techniques* and *character* ***S****imilarity assessments*. To tackle the third shortcomings and integrate LLMs with existing CSC models, we introduce a prudently replacement strategy (**SCP**), which utilizes the ***S****entence from existing models outputs*, ***C****andidates from LLMs aligned outputs* and *calculates* ***P****robability for potential candidates*, only replace the most likely wrong characters to prevent over-correction. Collectively, ARM bolster the performance of existing CSC models and overcoming the aforementioned limitations of LLMs.

In summary, the contributions of our work can be summarized into four aspects:

- We have developed a feasible module for utilizing LLMs in CSC. To the best of our knowledge, LLMs has seldom been employed in other CSC studies, marking a big step toward integrating LLMs with CSC.
- We propose alignment method ERS and replacement strategy SCP to address the challenges posed by LLMs.
- We introduce a plug-and-play method ARM, which can be integrated with almost any existing CSC models without requiring retraining or fine-tuning.
- We conduct extensive experiments on widely used public datasets and achieve state-of-the-art performance. Additionally, detailed analyses further validate the effectiveness of our proposed module.

2 Related Work

Chinese Spelling Check, an important task in natural language processing, emerged in the 1990s and has increasingly attracted scholarly attention over the past decade. Initially, scholars manually devised rules tailored to types of errors to facilitate correction (Mangu and Brill, 1997; Jiang et al., 2012; Zhang et al., 2021a). Subsequently, researchers adopted statistical methods, utilizing large-scale corpora to both detect and correct textual inaccuracies (Liu et al., 2013; Xie et al., 2015; Zhang et al., 2019).

Recently, the advent of deep learning has dramatically influenced CSC, particularly with the widespread adoption of PLMs such as BERT (Kenton and Toutanova, 2019) and RoBERTa (Liu et al., 2019). Innovations extend to the integration of phonetic and visual character information in models. For instance, REALISE (Xu et al., 2021) employs ResNet (Cho et al., 2014) to extract the visual information of characters, acquires word-level and sentence-level phonetic information by GRU (He et al., 2016) and Transformer Blocks.

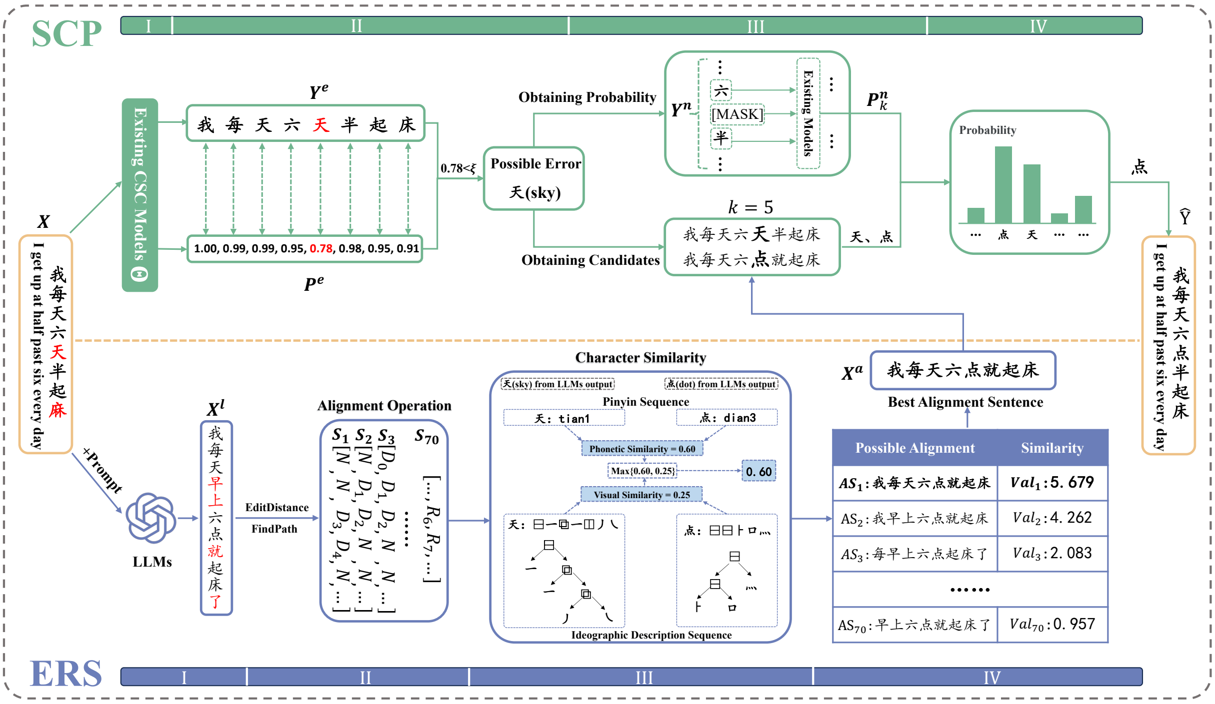

Figure 2: The architecture of ARM, which consists of alignment method ERS and replacement strategy SCP. Characters highlighted in red signify errors or redundancies and serial number corresponds to (§3). The bottom part illustrates how to use ERS to find the best alignment sentence "我每天六点就起床" among multiple choices. The top part reveals that existing models fails to correct the incorrect character "天" to the label "点" and how this error is successfully rectified by utilizing SCP.

Similarly, models like PLOME (Liu et al., 2021) and DCN (Wang et al., 2021)have also been developed to harness these information. Further advancements in CSC include the formulation of novel pretraining objectives and mask strategies. SCOPE (Li et al., 2022b) uses two parallel decoders with an adaptive weighting scheme and proposes fine granularity pinyin prediction task which predict the initial, final, and tone of pinyin. LEAD (Li et al., 2022c), CRASpell (Liu et al., 2022), and MFT (Wu et al., 2023) also design different training objectives or mask strategies. Additionally, some studies have sought to restructure model architectures to optimize correction processes. For example, SoftMask-BERT (Zhang et al., 2020) and MDCSPell (Zhu et al., 2022) explore the synergy between detection and correction networks. DR-CSC (Huang et al., 2023) breaks down CSC task into three subcomponents: detection, reasoning, and searching, which allows for more efficient leveraging of external Chinese linguistic knowledge.

Following the advent of LLMs, some researchers (Li et al., 2023; Dong et al., 2024) begin to explore the capabilities of LLMs in CSC. Their research pointed out that LLMs have many advantages in CSC tasks, such as better handling of complex CSC samples and better tolerance for errors. In addition, when evaluation criteria are adjusted, the performance of LLMs is found to be comparable to that of traditional models. Nevertheless, significant challenges still persist, such as the inability to constrain output length and the tendency to introduce unnecessary modifications. Therefore, our research aims to unveil LLMs potential and pioneer the integration of LLMs into CSC task.

3 Methodology

In this section, we commence with the formulation of the CSC task (§3.1). We then elaborate on our proposed ARM, depicted in Figure 2. Comprehensive details on the "Alignment Operation" are provided in the Appendix D. The introduction of our alignment method ERS is presented in §3.2, followed by a description of the replacement strategy SCP, which is shown in §3.3.

3.1 Problem Formulation

Chinese Spelling Check (CSC) can be formalized as the following task. Given a Chinese sentence $\boldsymbol{X} = \{x_1, x_2, \ldots, x_n\}$ of n characters that may include erroneous characters. We use $\boldsymbol{Y} = \{y_1, y_2, \ldots, y_n\}$ to represent the corresponding correct sentence. The sentence $\boldsymbol{X}$ and $\boldsymbol{Y}$ have the same length. The objective of CSC is to detect and correct the erroneous characters by generating a prediction $\hat{\boldsymbol{Y}} = \{\hat{y_1}, \hat{y_2}, \ldots, \hat{y_n}\}$ for the input $\boldsymbol{X}$, where $\hat{y_i}$ is the character predicted for x_i. The primary mission of CSC lies in accurately detecting the erroneous characters and predicting their correct counterparts in $\boldsymbol{Y}$.

3.2 Alignment Method

According to the above definition, CSC can be conceptualized as a sequence labeling task necessitating outputs of fixed length. However, due to the inherent properties of LLMs, despite efforts to design prompt to preserve the input length and specify output format, their outputs frequently deviate in length and format, occurring with a probability of 11%-27% as detailed in Appendix B. Consequently, to effectively use LLMs for CSC tasks, it is crucial to develop a method to align the input and output. Therefore, we propose the alignment method ERS, whose specific steps are as follows:

I: Get LLMs Response. First we combine $\boldsymbol{X}$ with $Prompt$, and then input it into LLMs to obtain the modified result $\boldsymbol{X^l}$, whose length is m:

$$\boldsymbol{X^l} = \mathrm{LLMs}(Prompt, \boldsymbol{X}). \tag{1}$$

II: Find Alignment Operations. Then, we find all possible alignment operations and obtain aligned sentences using those alignment operations. Initially, a dynamic programming algorithm $\mathrm{EditDistance}$ (shown in Appendix C) calculates the edit distance matrix $\boldsymbol{D}$ between $\boldsymbol{X^l}$ and $\boldsymbol{X}$. Subsequently, to identify all feasible transformations from $\boldsymbol{X^l}$ to $\boldsymbol{X}$, a recursive algorithm, $\mathrm{FindPath}$ (shown in Appendix D), is utilized. This algorithm enumerates all possible sequences of edit operations—insertions, deletions, and substitutions—that convert $\boldsymbol{X^l}$ into $\boldsymbol{X}$. The culmination of this process is the generation of the complete set of transformation sequences, collectively denoted as $\boldsymbol{S}$. The formulaic representation is as follows:

$$\boldsymbol{D} = \mathrm{EditDistance}(\boldsymbol{X^l}, \boldsymbol{X}), \tag{2}$$

$$\boldsymbol{S} = \mathrm{FindPath}(\boldsymbol{D}, \boldsymbol{X^l}, \boldsymbol{X}), \tag{3}$$

where $\boldsymbol{D} \in \mathbb{Z}^{(m+1)\times(n+1)}$ and $\boldsymbol{S}$ consists of p arrays, $p \in \mathbb{Z}$, each array represents a series of operations for an alignment approach.

By utilizing $\boldsymbol{S}$, we can restore $\boldsymbol{X^l}$ to a sentence of the same length as $\boldsymbol{X}$. In other words, $\boldsymbol{X}$ can be transformed into this restored sentence merely by replacement operations. Specifically, for replacement operations, no changes are made. For addition operations, the added part is removed. For deletion operations, the same location of $\boldsymbol{X}$ is referenced to fill in the deleted part. Ultimately, we obtain $\boldsymbol{AS} \in \mathbf{V}^{p\times n}$. $\boldsymbol{AS}$ possesses p aligned sentences and $\mathbf{V}$ is the vocabulary.

III: Calculate Character Similarity. To get the best alignment sentence, we propose a function ChSim that calculates the similarity between two characters by considering both their phonetic and visual similarities, and taking the maximum value of the two as the final similarity. Specifically, we draw on the work of Hong et al. (2019); Li et al. (2022a) to use the pinyin sequence of characters for phonetic information and the ideographic description sequence (IDS) for visual information. The phonetic and visual similarities are computed using the edit distance and the results are then inverted and normalized to yield the final similarity score. The specific formula is as follows:

$$s_1 = 1 - \frac{\mathrm{ED}(\boldsymbol{py}^a, \boldsymbol{py}^b)}{\max\{|\boldsymbol{py}^a|, |\boldsymbol{py}^b|\}}, \tag{4}$$

$$s_2 = 1 - \frac{\mathrm{ED}(\boldsymbol{ids}^a, \boldsymbol{ids}^b)}{\max\{|\boldsymbol{ids}^a|, |\boldsymbol{ids}^b|\}}, \tag{5}$$

$$\mathrm{ChSim}(a, b) = \max\{s_1, s_2\}, \tag{6}$$

where a and b denote two characters, s_1 and $s_2 \in \mathbb{R}$, $\boldsymbol{ids}$ and $\boldsymbol{py}$ denotes the IDS and pinyin sequence of the characters. The function ED merely return the edit distance between two sequences instead of returning the entire matrix like equation (2), which can refer to Algorithm 1.

IV: Choose Best Alignment Sentence. Finally, we select the sentence from $\boldsymbol{AS}$ that exhibits the highest similarity to sentence $\boldsymbol{X}$, deeming it the best alignment sentence. To determine the similarity of between two sentence, we calculate the similarity between each character pair and then sum these values. The formulas are as follows:

$$\boldsymbol{Val}_j = \sum_{i=1}^{n} \mathrm{ChSim}(\boldsymbol{AS}_{j,i}, x_i), \tag{7}$$

$$\boldsymbol{X^a} = \boldsymbol{AS}_{\arg\max_j \boldsymbol{Val}_j}, \tag{8}$$

where $j \in [1, 2, \cdots, p]$, $\boldsymbol{Val}_j$ represents the similarity between the j-th sentence in $\boldsymbol{AS}$ and $\boldsymbol{X}$, and $\boldsymbol{AS}_{j,i}$ represents the i-th character in the j-th sentence in $\boldsymbol{AS}$. Eventually, we get best alignment sentence $\boldsymbol{X^a}$, which will be used in the replacement strategy SCP.

3.3 Replacement Strategy

Current CSC models predominantly utilize non-autoregressive PLMs. These models transform the final hidden vector into a probability distribution using a $\mathrm{softmax}$ function. They compute the probability of each word in $\mathbf{V}$ for a certain position and select the character with the highest probability as the output for that position. Typically, a high maximum probability signifies their confidence in character selection, while a low maximum probability indicates uncertainty. Therefore, the magnitude of the maximum probability can serve as an indicator of potential errors.

Based on the above discussion, we propose the replacement strategy SCP, designed to leverage the aligned sentence from ERS method and the output probabilities from existing CSC models Θ to correct potential errors generated by Θ. Its specific steps are as follows:

I: Get Original Revised Sentence. we initially obtain the modified sentence $\boldsymbol{Y^e}$, and the corresponding probability $\boldsymbol{P^e}$ from Θ:

$$\boldsymbol{Y^e}, \boldsymbol{P^e} = \Theta(\boldsymbol{X}), \tag{9}$$

$$\hat{\boldsymbol{Y}} = \boldsymbol{Y^e}, \tag{10}$$

where $\boldsymbol{Y^e} \in \mathbf{V}^n$, $\boldsymbol{P^e} \in \mathbb{R}^{n\times r}$, and $|\mathbf{V}| = r$, r is the size of vocabulary. From §3.1, $\hat{\boldsymbol{Y}} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_n}\}$ is the final output.

II: Select Possible Error. Then we set a hyper-parameter threshold ξ and $0 < \xi < 1$. If the max probability of position k is less than ξ, which can also be formalized as $||\boldsymbol{P^e}_k||_\infty < \xi$, it suggests a potential error at k-th location.

III: Obtain Probability and Candidate. Subsequently, we mask the identified position and recall Θ to calculate the probability for k-th position, which can formulate as follows:

$$\boldsymbol{Y^n} = [\cdots, \boldsymbol{Y^e}_{k-1}, [\text{MASK}], \boldsymbol{Y^e}_{k+1}, \cdots], \tag{11}$$

$$\boldsymbol{P^n} = \Theta(\boldsymbol{Y^n}), \tag{12}$$

where $\boldsymbol{P^n} \in \mathbb{R}^{n\times r}$ whose meaning is similar to $\boldsymbol{P^e}$. $\boldsymbol{P^n}_k \in \mathbb{R}^r$ and is the probability of character at the k-th position.

IV: Replace Possible Error. We select the character that exhibits the highest probability value in $\boldsymbol{P^n}_k$ among the character in the best alignement sentence $\boldsymbol{X^a}$ and sentence $\boldsymbol{Y^e}$ modified by Θ, to determine the final output. The formula is expressed as follows:

$$i, j = \mathrm{ID}(\boldsymbol{Y^e}_k), \mathrm{ID}(\boldsymbol{X^a}_k), \tag{13}$$

$$\hat{y_k} = \begin{cases} \boldsymbol{Y^e}_k & \boldsymbol{P^n}_{ki} \geq \boldsymbol{P^n}_{kj}, \\ \boldsymbol{X^a}_k & \text{Otherwise.} \end{cases} \tag{14}$$

where $\mathrm{ID}(\cdot)$ is a function that assigns each character to a number based on $\boldsymbol{V}$. Those steps facilitate the substitution at position k where the confidence is low. To generate the final output, equations (11) (12) (13) (14) is reiterated for each position in modified sentence where the probability falls below the threshold ξ, culminating in applying replacement strategy to the entire sentence.

However, owing to variations in pre-training and fine-tuning techniques, certain models, like Li et al. (2022b), do not employ the "[MASK]" token during training and thus fail to comprehend this token. To solve this problem, we propose an alternative approach, which is shown in Appendix E.

4 Experiments and Results

4.1 Experimental Setup

4.1.1 Dataset

In this study, the commonly used datasets SIGHAN13 (Wu et al., 2013b), SIGHAN14 (Yu et al., 2014), and SIGHAN15 (Tseng et al., 2015), along with the W271K (Wang et al., 2018) are used in training process. Our proposed module is evaluated on SIGHAN13/14/15 test sets like previous work (Liu et al., 2021; Xu et al., 2021; Li et al., 2022d; Zhang et al., 2022a; Wei et al., 2023; Liang et al., 2023). Furthermore, considering the SHIGHAN is in traditional Chinese, following prior research (Li et al., 2022d; Liang et al., 2023), we utilized the OpenCC[1] tools to convert it to simplified Chinese.

4.1.2 Baseline Model

To evaluate the performance of ARM, we selected several advanced CSC models to compare: **FASpell** (Hong et al., 2019) utilizes a confidence-similarity decoder to filter out visually or phonologically irrelevant candidates. **SoftMasked-BERT** (Zhang et al., 2020) uses Bi-GRU to detect errors and uses BERT to correct those errors. **REALISE** (Xu et al., 2021) employs ResNet to extract

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

Dataset	Model	Detection-level P	Detection-level R	Detection-level F	Correction-level P	Correction-level R	Correction-level F
SIGHAN13	FASpell (Hong et al., 2019)	76.2	63.2	69.1	73.1	60.5	66.2
	REALISE (Xu et al., 2021)	88.6	82.5	85.4	87.2	81.2	84.1
	DORM (Liang et al., 2023)	87.9	83.7	85.8	86.8	82.7	84.7
	DR-CSC (Huang et al., 2023)	88.5	83.7	86.0	**87.7**	83.0	**85.3**
	SoftMask-BERT (Zhang et al., 2020)†	85.2	78.0	81.4	83.8	76.8	80.1
	SoftMask-BERT+ARM	85.9↑	79.5↑	82.6↑	84.6↑	78.2↑	81.3↑
	MDCSPell (Zhu et al., 2022)†	85.7	78.5	82.0	84.6	77.5	80.9
	MDCSPell+ARM	86.4↑	79.5↑	82.8↑	85.5↑	78.6↑	81.9↑
	SCOPE (Li et al., 2022c)	87.4	83.4	85.4	86.3	82.4	84.3
	SCOPE+ARM	**88.7**↑	**84.1**↑	**86.3**↑	87.6↑	**83.1**↑	**85.3**↑
SIGHAN14	FASpell (Hong et al., 2019)	61.0	53.5	57.0	59.4	52.0	55.4
	REALISE (Xu et al., 2021)	67.8	71.5	69.6	66.3	70.0	68.1
	DORM (Liang et al., 2023)	69.5	73.1	71.2	68.4	71.9	70.1
	DR-CSC (Huang et al., 2023)	70.2	73.2	71.7	**69.3**	72.3	70.7
	SoftMask-BERT (Zhang et al., 2020)†	69.6	69.6	69.6	68.5	68.5	68.5
	SoftMask-BERT+ARM	70.4↑	71.3↑	70.9↑	**69.3**↑	70.2↑	69.7↑
	MDCSPell (Zhu et al., 2022)†	66.2	66.5	66.3	64.2	64.6	64.4
	MDCSPell+ARM	67.3↑	68.8↑	68.1↑	65.4↑	66.9↑	66.2↑
	SCOPE (Li et al., 2022c)	70.1	73.1	71.6	68.6	71.5	70.1
	SCOPE+ARM	**71.2**↑	**75.0**↑	**73.1**↑	69.2↑	**73.0**↑	**71.1**↑
SIGHAN15	FASpell (Hong et al., 2019)	67.6	60.0	63.5	66.6	59.1	62.6
	REALISE (Xu et al., 2021)	77.3	81.3	79.3	75.9	79.9	77.8
	DORM (Liang et al., 2023)	77.9	84.3	81.0	76.6	82.8	79.6
	DR-CSC (Huang et al., 2023)	**82.9**	84.8	83.8	**80.3**	82.3	**81.3**
	SoftMask-BERT (Zhang et al., 2020)†	75.5	79.2	77.3	74.1	77.8	75.9
	SoftMask-BERT+ARM	76.4↑	80.9↑	78.6↑	74.7↑	79.0↑	76.8↑
	MDCSPell (Zhu et al., 2022)†	76.3	79.6	77.9	75.2	78.5	76.8
	MDCSPell+ARM	76.4↑	81.3↑	78.8↑	75.2	80.0↑	77.5↑
	SCOPE (Li et al., 2022c)	81.1	84.3	82.7	79.2	82.3	80.7
	SCOPE+ARM	82.3↑	**86.1**↑	**84.1**↑	79.5↑	**83.1**↑	**81.3**↑

Table 1: The performance of ARM and baseline models. X+ARM indicates the integration of ARM with a baseline model X. The highest scores for specific metrics are highlighted in bold. The symbol '↑' denotes an improvement in performance following the integration of ARM with the baseline models, and '†' signifies that the presented data are outcomes of self-training and not directly extracted from existing literature.

visual information and fuses it with phonetic information and semantic. **SCOPE** (Li et al., 2022b) researches the adaptivity and granularity of pronunciation prediction and design a iterative correction strategy. **MDCSPell** (Zhu et al., 2022) integrates the hidden states from the detection and correction modules using a late fusion strategy to minimize the misleading impact of typos. **DORM** (Liang et al., 2023) introduces a pinyin-to-character prediction task with a separation mask and a self-distillation module to ensure that the model does not overfit on phonetic features. **DR-CSC** (Huang et al., 2023) breaks down CSC task into three sub-components: detection, reasoning, and searching, which is efficient of using external knowledge.

4.1.3 Evaluation Metrics

Referring to the processing and evaluation methodologies employed in prior research (Xu et al., 2021; Li et al., 2022b; Zhang et al., 2022b; Li et al., 2022a; Liang et al., 2023), our test approach is delineated as follows: We utilize sentence-level evaluation metrics that impose more rigorous standards than character-level metrics. Specifically, we assess the model's capabilities in error detection level and correction level through three key indicators: **Precision**, **Recall**, and **F1 scores**. Additionally, in SIGHAN13, because of a lot of mixed usage of "的", "地", "得" which are easily confused auxiliary words that modify adjectives, nouns, and verbs. We remove all detected and corrected "的", "地",

Model		CAR	COT	ENC	GAM	MEC	NEW	NOV
GPT-3.5-Turbo	D	22.1	27.4	30.4	18.1	33.7	17.5	16.8
	C	18.5	22.0	25.8	14.1	27.9	13.1	11.7
SoftMask	D	39.2	57.3	39.3	17.1	36.4	39.3	18.8
	C	31.6	44.2	31.7	12.1	29.8	32.3	15.5
SoftMask+ARM	D	40.6(↑1.4)	58.3(↑1.0)	40.9(↑1.6)	18.7(↑1.6)	38.7(↑2.3)	41.0(↑1.7)	19.7(↑0.9)
	C	33.2(↑1.6)	45.5(↑1.3)	33.7(↑2.0)	13.9(↑1.8)	32.4(↑2.6)	34.4(↑2.1)	16.5(↑1.0)
MDCSPell	D	41.5	61.8	41.0	19.3	37.0	42.5	17.9
	C	34.1	49.2	32.8	14.8	29.5	34.4	14.3
MDCSPell+ARM	D	44.3(↑2.8)	64.4(↑2.6)	42.9(↑1.9)	19.6(↑0.3)	40.0(↑3.6)	44.2(↑1.7)	19.0(↑1.1)
	C	37.1(↑3.0)	52.7(↑3.5)	35.2(↑2.4)	15.3(↑0.5)	33.0(↑3.5)	36.4(↑2.0)	15.6(↑1.3)

Table 2: The performance of GPT-3.5-Turbo and some models on the LEMON datasets. CAR, COT, ENC, GAM, MEC, NEW, and NOV are seven distinct fields. "D" and "C" indicate the detection-level and the correction-level F1-index. SoftMask means SoftMask-BERT.

"得" from the model output before evaluation.

4.1.4 Implementation Details

In the experiments, we employ PyTorch to implement the proposed ARM, namely SoftMasked-BERT and MDCSPell. The initialization weights for these models are sourced from a GitHub repository[2], and they are fine-tuned using the MFT (Wu et al., 2023). We set the maximum sentence length to 512 to accommodate all sentence length and ξ to 0.9. The training is conducted with a batch size of 16, using the AdamW optimizer and a learning rate of 1×10^{-5}. Additionally, for training SCOPE, we utilize code and parameters from the official SCOPE repository[3]. All experiments are conducted on a single GeForce RTX 4090.

In terms of selecting LLMs, we utilized the interface provided by OpenAI to access the GPT-3.5-Turbo[4] and keep all parameters such as temperature, topn, etc. as default. Additionally, details about the prompt used are comprehensively outlined in Appendix F.

4.2 Experimental Results

Table 1 illustrates the effectiveness of augmenting existing CSC models with ARM, as evidenced by enhanced performance metrics such as F1 scores. The models enhanced include SoftMask-BERT, MDCSPell, and SCOPE, which all show improvements across the test datasets. For instance, the integration of ARM with SoftMask-BERT resulted in F1 scores increases of 1.2%, 1.2%, and 0.9% across three respective datasets. Similarly, MDCSPell, when augmented with ARM, experienced improvements of 1.0%, 1.8%, and 0.7%, and SCOPE with ARM achieved gains of 1.0%, 1.0%, and 0.6% in each dataset. These results confirm the proposed ARM model's capability to enhance the accuracy and efficiency of existing systems.

Furthermore, the combination of SCOPE and ARM achieves state-of-the-art performance across three datasets, thereby underscoring ARM's competitive edge within CSC task.

4.3 Analysis and Discussion

4.3.1 Performance of ARM on mutil-domain datasets

In this part, we evaluate the model described in § 4.1.2 on a multi-domain dataset LEMON (Wu et al., 2023) without extra training, to verify the ability of ARM on multi-domain datasets. The LEMON dataset encompasses over 20,000 sentences drawn from seven distinct domains, including car (CAR), contract (COT), encyclopedia (ENC), game(GAM), medical care (MEC), news (NEW) and novel (NOV).

From Table 2, we can draw the following conclusions. First, the traditional model, despite lacking domain-specific training, outperforms GPT-3.5-Turbo, highlighting limitations within LLMs. Furthermore, integrating the traditional model with ARM yields substantial performance gains, indicating that ARM effectively transfers domain knowledge to the traditional model while addressing the limitations inherent to LLMs.

These findings underscore the complementary

[2] https://github.com/brightmart/roberta_zh
[3] https://github.com/jiahaozhenbang/SCOPE
[4] https://platform.openai.com/

Dataset	Model	Ori		Ran		Tru		Ali	
		D	C	D	C	D	C	D	C
SIGHAN13	SoftMask-BERT+ARM	74.3	72.0	74.3	72.0	76.1	74.1	76.8	74.8
	MDCSPell+ARM	75.6	74.1	75.6	74.1	76.3	75.0	77.4	76.1
SIGHAN14	SoftMask-BERT+ARM	64.5	63.3	64.5	63.3	64.8	63.6	65.5	64.3
	MDCSPell+ARM	60.7	58.4	60.7	58.4	62.9	60.7	63.3	61.1
SIGHAN15	SoftMask-BERT+ARM	67.8	66.8	67.8	66.8	68.2	67.3	68.2	67.3
	MDCSPell+ARM	71.8	71.3	71.8	71.3	72.4	71.4	73.9	73.0

Table 3: The impact of different candidates provision methods on replacement strategy and F1 scores testing in sentences of varying lengths in the LLMs responses. “Ori” serves as the benchmark. “Tru”, “Ran”, and “Ali” denote three distinct approaches to supplying candidates: “Ran” refers to candidates obtained from a random Chinese character; “Tru” involves candidates derived through simple truncation and padding of sentences; and “Ali” represents candidates sourced from the ERS. “D” and “C” indicate the detection-level and the correction-level.

relationship between LLMs and traditional models, collectively enhancing performance and further demonstrate the great potential of LLMs in CSC.

4.3.2 Rigorousness of Replacement Strategy

In this part, we demonstrate the rigor of our proposed replacement strategy SCP by investigating the impact of different candidates provision methods. We analyzed the performance using sentences from the SIGHAN test set, whose length is changed by LLMS. This analysis focused on the F1 scores, and the results are presented in Table 3.

From the experimental data, we find firstly, while the candidates generated by method “Ran” are of lower quality, there is no reduction in the F1 scores compared to benchmark “Ori”. Secondly, although the candidates from method “Tru” are not of high quality, they contribute to a little enhancement in experimental outcomes. Finally, method “Ali” stands out by generating high-quality candidates, which substantially improve the F1 scores.

According to the above, existing models replace characters based on probability assessments; it only substitutes an original character when the candidate’s probability exceeds that of the original. Consequently, the quality of the replacement candidates is crucial: high-quality candidates enhance the model’s performance, while low-quality candidates do not adversely affect it. This proves the rigor of the replacement strategy SCP. Additionally, the alignment method ERS demonstrates superior performance compared to other approaches, highlighting its ability to generate higher quality candidates and its overall effectiveness.

4.3.3 Analysis of Alignment Method

In this part, we demonstrate the effectiveness and competitiveness of our alignment method ERS. Our analysis employs three distinct processing techniques on the responses generated by the LLMs. The first approach was to analyse the responses in their original, unaltered form. The second method employs truncation and padding to adjust sentence lengths. The third method applies our proposed best alignment method ERS. To assess the performance of these methods, we calculate F1 scores at both the detection level and correction level using the SIGHAN dataset.

The results of these evaluations are presented in Table 4. From these data, we can draw the following conclusions. First, compared with Table 1 the performance of direct responses of LLMs is markedly inadequate, demonstrating significant deficiencies in CSC. Second, direct truncation and padding offer only limited improvement on the F1 scores. Third, the implementation of our method, ERS, significantly enhances the F1 scores, with improvements ranging from 1.9% to 9.3%, proving the effectiveness of our method.

4.3.4 Case Study

To illustrate how the alignment method ERS and replacement strategy SCP can effectively help correction and address the limitations of LLMs, we selsct several cases from the SIGHAN test set, which is shown in Table 5. In the first example, the existing CSC model erroneously substituted “清昕 (limpid dawn)” with “清澈 (pellucidly)”. Simultaneously, LLMs correctly substituted that “清昕 (limpid dawn)” by “清晰 (clearly)”, but inaccurately changed “飞翔 (fly)” to “飞舞 (dance in the

Dataset		Ori	Tru	Ali
13 Train	D	52.4	54.0(↑1.6)	59.3(↑6.9)
	C	43.0	44.1(↑1.1)	48.2(↑5.2)
13 Test	D	46.0	46.1(↑0.1)	51.9(↑5.9)
	C	36.4	36.5(↑0.1)	41.3(↑3.9)
14 Train	D	37.6	38.0(↑0.4)	45.3(↑7.7)
	C	30.1	30.3(↑0.2)	35.5(↑5.4)
14 Test	D	30.1	30.6(↑0.5)	39.4(↑9.3)
	C	25.4	26.0(↑0.6)	32.0(↑6.6)
15 Train	D	45.0	45.0(↑0.0)	48.8(↑3.8)
	C	38.1	38.2(↑0.1)	41.0(↑1.9)
15 Test	D	44.8	44.8(↑0.0)	49.0(↑4.2)
	C	37.4	37.4(↑0.0)	40.7(↑3.3)

Table 4: The F1 scores for various processing methods applied to LLMs answers on the SIGHAN dataset.

air)" and introduced the unnecessary character "以 (can)". Utilizing the alignment method, the character "以 (can)" was successfully removed, although "飞舞 (dance in the air)" was still not corrected. In replacement step, only the low-probability character "昕 (dawn)" was replaced with "晰 (clearly)" and the redundant corrected character "舞 (dance)" is not replaced. Similarly, in the second example, the existing CSC model failed to correct the error character "郎 (man)" and LLMs added an unnecessary character "得 (a function word)", but ARM ultimately corrected "郎 (man)" to "朗 (bright)" without any other modification.

Case1:	
Input:	终于可清昕望见喜鹊飞翔。
CSCModel:	终于可清澈望见喜鹊飞翔。
LLMs:	终于可以清晰望见喜鹊飞舞。
Aligned:	终于可清晰望见喜鹊飞舞。
ARM:	终于可清晰望见喜鹊飞翔。
Translation:	Finally, I clearly see the magpies flying.
Case2:	
Input:	我好像真的变开郎了。
CSCModel:	我好像真的变开郎了。
LLMs:	我好像真的变得开朗了。
Aligned:	我好像真的变开朗了。
ARM:	我好像真的变开朗了。
Translation:	I seem to really become more sanguine.

Table 5: Examples from SIGHAN show how to correct sentence by existing CSC model, LLMs and the proposed ARM. Incorrect and redundant characters are highlighted in red and green, while correct counterparts are indicated in blue.

5 Conclusion

We introduces ARM, a novel module designed to ameliorate critical deficiencies in LLMs when applied to CSC task. ARM encompass two principal approaches: the alignment method ERS and replacement strategies SCP. ERS processes sentences where outputs do not match inputs in length, and enhances the alignment of LLMs output with the original sentence in terms of length and similarity. Concurrently, SCP rigorously assesses the appropriateness of candidates provided by LLMs, determining whether they should supplant the outputs from existing CSC models. By incorporating these approaches with current CSC models, ARM have demonstrated superior performance, achieving state-of-the-art results across three SIGHAN datasets, thereby demonstrating the module's effectiveness and competitiveness.

6 Limitations

The limitations of this paper is twofold. Firstly, the dataset utilized is relatively dated and limited in scope, and it contains numerous errors. Consequently, the full capabilities of LLMs in the CSC task waiting further exploration and this study merely presents a viable approach to employing LLMs. Secondly, in CSC task, spelling errors do not always necessitate a singular correct modification; multiple valid corrections can exist. Thus, developing more robust evaluation metrics for CSC represents a valuable avenue for future research.

Acknowledgements

This research was partial supported by grants from the National Key Research, Development Program of China (Grant No. 2021YFF0901000), the National Natural Science Foundation of China (No. U20A20229), Anhui Provincial Natural Science Foundation (No. 2308085QF229), and the Fundamental Research Funds for the Central Universities (No. WK2150110034).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Kyunghyun Cho, Bart van Merriënboer, Çağlar Gulçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Ming Dong, Yujing Chen, Miao Zhang, Hao Sun, and Tingting He. 2024. Rich semantic knowledge enhanced large language models for few-shot chinese spell checking. *arXiv preprint arXiv:2403.08492*.

Jianfeng Gao, Chris Quirk, et al. 2010. A large scale ranker-based system for search query spelling correction. In *The 23rd international conference on computational linguistics*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu. 2019. Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 160–169.

Haojing Huang, Jingheng Ye, Qingyu Zhou, Yinghui Li, Yangning Li, Feng Zhou, and Hai-Tao Zheng. 2023. A frustratingly easy plug-and-play detection-and-reasoning module for chinese spelling check. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11514–11525.

Ying Jiang, Tong Wang, Tao Lin, Fangjie Wang, Wenting Cheng, Xiaofei Liu, Chenghui Wang, and Weijian Zhang. 2012. A rule based chinese spelling and grammar detection system utility. In *2012 International Conference on System Science and Engineering (ICSSE)*, pages 437–440. IEEE.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

Fangfang Li, Youran Shan, Junwen Duan, Xingliang Mao, and Minlie Huang. 2022a. Wspeller: Robust word segmentation for enhancing chinese spelling check. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1179–1188.

Jiahao Li, Quan Wang, Zhendong Mao, Junbo Guo, Yanyan Yang, and Yongdong Zhang. 2022b. 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.

Yinghui Li, Haojing Huang, Shirong Ma, Yong Jiang, Yangning Li, Feng Zhou, Hai-Tao Zheng, and Qingyu Zhou. 2023. On the (in) effectiveness of large language models for chinese text correction. *arXiv preprint arXiv:2307.09007*.

Yinghui Li, Shirong Ma, Qingyu Zhou, Zhongli Li, Li Yangning, Shulin Huang, Ruiyang Liu, Chao Li, Yunbo Cao, and Haitao Zheng. 2022c. Learning from the dictionary: Heterogeneous knowledge guided fine-tuning for chinese spell checking. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 238–249.

Yinghui Li, Qingyu Zhou, Yangning Li, Zhongli Li, Ruiyang Liu, Rongyi Sun, Zizhen Wang, Chao Li, Yunbo Cao, and Hai-Tao Zheng. 2022d. 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.

Zihong Liang, Xiaojun Quan, and Qifan Wang. 2023. Disentangled phonetic representation for chinese spelling correction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13509–13521.

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.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2024a. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36.

Linfeng Liu, Hongqiu Wu, and Hai Zhao. 2024b. Chinese spelling correction as rephrasing language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18662–18670.

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.

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.

Xiaodong Liu, Kevin Cheng, Yanyan Luo, Kevin Duh, and Yuji Matsumoto. 2013. A hybrid chinese spelling correction using language model and statistical machine translation with reranking. In *Proceedings of the seventh SIGHAN workshop on chinese language processing*, pages 54–58.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lidia Mangu and Eric Brill. 1997. Automatic rule acquisition for spelling correction. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 187–194.

Bruno Martins and Mário J Silva. 2004. Spelling correction for search engine queries. In *Advances in Natural Language Processing: 4th International Conference, EsTAL 2004, Alicante, Spain, October 20-22, 2004. Proceedings 4*, pages 372–383. Springer.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

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.

Baoxin Wang, Wanxiang Che, Dayong Wu, Shijin Wang, Guoping Hu, and Ting Liu. 2021. Dynamic connected networks for chinese spelling check. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2437–2446.

Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. A hybrid approach to automatic corpus generation for chinese spelling check. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2517–2527.

Xiao Wei, Jianbao Huang, Hang Yu, and Qian Liu. 2023. Ptcspell: Pre-trained corrector based on character shape and pinyin for chinese spelling correction. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6330–6343.

Hongqiu Wu, Shaohua Zhang, Yuchen Zhang, and Hai Zhao. 2023. Rethinking masked language modeling for chinese spelling correction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10743–10756.

Jian-cheng Wu, Hsun-wen Chiu, and Jason S Chang. 2013a. Integrating dictionary and web n-grams for chinese spell checking. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 18, Number 4, December 2013-Special Issue on Selected Papers from ROCLING XXV*.

Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013b. Chinese spelling check evaluation at sighan bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42.

Weijian Xie, Peijie Huang, Xinrui Zhang, Kaiduo Hong, Qiang Huang, Bingzhou Chen, and Lei Huang. 2015. Chinese spelling check system based on n-gram model. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 128–136.

Heng-Da Xu, Zhongli Li, Qingyu Zhou, Chao Li, Zizhen Wang, Yunbo Cao, He-Yan Huang, and Xian-Ling Mao. 2021. Read, listen, and see: Leveraging multimodal information helps chinese spell checking. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 716–728.

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.

Kai Zhang, Qi Liu, Zhenya Huang, Mingyue Cheng, Kun Zhang, Mengdi Zhang, Wei Wu, and Enhong Chen. 2022a. Graph adaptive semantic transfer for cross-domain sentiment classification. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1566–1576.

Kai Zhang, Qi Liu, Hao Qian, Biao Xiang, Qing Cui, Jun Zhou, and Enhong Chen. 2021a. Eatn: An efficient adaptive transfer network for aspect-level sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):377–389.

Kai Zhang, Hao Qian, Qing Cui, Qi Liu, Longfei Li, Jun Zhou, Jianhui Ma, and Enhong Chen. 2021b. Multi-interactive attention network for fine-grained feature learning in ctr prediction. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 984–992.

Kai Zhang, Hefu Zhang, Qi Liu, Hongke Zhao, Hengshu Zhu, and Enhong Chen. 2019. Interactive attention transfer network for cross-domain sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5773–5780.

Kai Zhang, Kun Zhang, Mengdi Zhang, Hongke Zhao, Qi Liu, Wei Wu, and Enhong Chen. 2022b. Incorporating dynamic semantics into pre-trained language model for aspect-based sentiment analysis. *arXiv preprint arXiv:2203.16369*.

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

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

A LLMs Test Results in SIGHAN

Table 6 presents the performance of the LLMs GPT-3.5-Turbo and ERNIE-3.5-8K[5] from BaiDu on three SIGHAN test datasets in CSC task. The results indicate that relying solely on the LLMs yields significantly poorer outcomes compared to those achieved by existing models based on PLMs, which suggests that LLMs may possess inherent limitations that critically undermine their effectiveness in the CSC task.

Model	Detection-level P	Detection-level R	Detection-level F	Correction-level P	Correction-level R	Correction-level F
GPT 13	45.2	62.3	52.4	37.1	51.2	43.0
GPT 14	37.7	37.6	37.6	30.1	30.1	30.1
GPT 15	45.7	44.3	45.0	38.7	37.6	38.1
ERNIE 13	18.0	14.0	15.7	16.9	13.2	14.8
ERNIE 14	5.5	8.5	6.7	5.0	7.8	6.0
ERNIE 15	15.3	25.1	19.0	13.3	22.0	16.6

Table 6: Experimental results of GPT-3.5-Turbo and ERNIE-3.5-8K on the SIGHAN test datasets. "13", "14", and "15" correspond to the "SIGHAN13", "SIGHAN14", and "SIGHAN15" respectively.

B LLMs Outputs Stability

We utilized the GPT-3.5 Turbo interface to input the relevant rules of the CSC task as a prompt, along with the sentence from CSC benchmark (Wu et al., 2013b; Yu et al., 2014; Tseng et al., 2015), to obtained the corrected sentence. Additionally, we recorded the difference in length between each corrected sentence and its corresponding original sentence. To mitigate the effects of randomness, each sentence was inputted four times. The results are presented in Table 7.

C Edit Distance Algorithm

The algorithm EditDistance calculates the edit distance between the LLM's response $\boldsymbol{Y^l}$ and the input sentence $\boldsymbol{X}$, which is the minimum number of operations required to transform $\boldsymbol{Y^l}$ into $\boldsymbol{X}$. These operations include addition, deletion, and replacement. It finally produces an output matrix $\boldsymbol{D}$, where $\boldsymbol{D}_{i,j}$ represents the edit distance between the first i words of Y^l and the first j words of $\boldsymbol{X}$. The detailed pseudocode is presented in Algorithm 1.

[5]https://cloud.baidu.com/

Dataset	Total	Unequal	Probability
13 Train	2,800	323	0.12
13 Test	4,000	742	0.19
14 Train	13,748	3,747	0.27
14 Test	4,248	1,001	0.24
15 Train	9,356	1,788	0.19
15 Test	4,400	777	0.18

Table 7: The probability of answers generated by LLMs differ in length from the input. Specifically, "13", "14", and "15" correspond to the "SIGHAN13", "SIGHAN14", and "SIGHAN15". "Total" refers to the total number of input, and "Unequal" refers to the number of responses that are not equal to the input length.

Algorithm 1 EditDistance

Input: $\boldsymbol{Y^l}$ and $\boldsymbol{X}$;
Output: $\boldsymbol{D}$;
$m \leftarrow$ length of $\boldsymbol{Y^l}$
$n \leftarrow$ length of $\boldsymbol{X}$
Create $\boldsymbol{D} \in \mathbb{Z}^{(m+1)\times(n+1)}$
for $i = 1$ to m **do**
    $\boldsymbol{D}_{i,0} \leftarrow i$
end for
for $j = 1$ to n **do**
    $\boldsymbol{D}_{0,j} \leftarrow j$
end for
for $i = 1$ to m **do**
    for $j = 1$ to n **do**
        if $\boldsymbol{Y^l}_{i-1} = \boldsymbol{X}_{j-1}$ **then**
            $cost \leftarrow 0$
        else
            $cost \leftarrow 1$
        end if
        $\boldsymbol{D}_{i,j} \leftarrow \min($
    $\boldsymbol{D}_{i-1,j} + 1,$
    $\boldsymbol{D}_{i,j-1} + 1,$
    $\boldsymbol{D}_{i-1,j-1} + cost)$
    end for
end for
return $\boldsymbol{D}$

D Recursive Algorithm for Finding Paths

The algorithm FindPath employs $\boldsymbol{D}$ from the EditDistance algorithm to identify all feasible operations transforming $\boldsymbol{Y^l}$ into $\boldsymbol{X}$. The result, labeled as $\boldsymbol{S}$, contains all potential methods of transformation. Specifically, $R_{i,j}$ denotes the replacement of the i-th character in $\boldsymbol{Y^l}$ with the j-th character in $\boldsymbol{X}$, I_j represents the insertion of the j-th character of $\boldsymbol{X}$ at the j-th position in $\boldsymbol{Y^l}$, D_i indicates the deletion of the i-th character in $\boldsymbol{Y^l}$, and N signifies no operation. The specific pseudocode is presented in Algorithm 2.

Algorithm 2 FindPath

Input: $\boldsymbol{Y^l}$, $\boldsymbol{X}$ and $\boldsymbol{D}$;
Output: $\boldsymbol{S}$;
function FINDALLPATHS(i, j)
  if $i = 0\ \ \&\ \ j = 0$ **then**
    return [[]]
  end if
  $\boldsymbol{p}, \boldsymbol{sub\_p} \leftarrow [\,], [\,]$
  if $i > 0\ \ \&\ \ \boldsymbol{D}_{i,j} = \boldsymbol{D}_{i-1,j} + 1$ **then**
    $\boldsymbol{sub\_p} \leftarrow$ FINDALLPATHS$(i-1, j)$
    for $\boldsymbol{path} \in \boldsymbol{sub\_p}$ **do**
      $\boldsymbol{p} \leftarrow \boldsymbol{p} \bigcup \{\boldsymbol{path} \bigcup \{D_{i-1}\}\}$
    end for
  end if
  if $j > 0\ \ \&\ \ \boldsymbol{D}_{i,j} = \boldsymbol{D}_{i,j-1} + 1$ **then**
    $\boldsymbol{sub\_p} \leftarrow$ FINDALLPATHS$(i, j-1)$
    for $path \in \boldsymbol{sub\_p}$ **do**
      $\boldsymbol{p} \leftarrow \boldsymbol{p} \bigcup \{\boldsymbol{path} \bigcup \{I_{j-1}\}\}$
    end for
  end if
  if $i > 0\ \ \&\ \ j > 0\ \ \&\ \ \boldsymbol{D}_{i,j} = \boldsymbol{D}_{i-1,j-1} + 1$ **then**
    $\boldsymbol{sub\_p} \leftarrow$ FINDALLPATHS$(i-1, j-1)$
    for $path \in \boldsymbol{sub\_p}$ **do**
      $\boldsymbol{p} \leftarrow \boldsymbol{p} \bigcup \{\boldsymbol{path} \bigcup \{R_{(i-1)(j-1)}\}\}$
    end for
  end if
  if $i > 0\ \ \&\ \ j > 0\ \ \&\ \ \boldsymbol{D}_{i,j} = \boldsymbol{D}_{i-1,j}$ **then**
    $\boldsymbol{sub\_p} \leftarrow$ FINDALLPATHS$(i-1, j)$
    for $path \in \boldsymbol{sub\_p}$ **do**
      $\boldsymbol{p} \leftarrow \boldsymbol{p} \bigcup \{\boldsymbol{path} \bigcup \{N\}\}$
    end for
  end if
  return $paths$
end function
$\boldsymbol{S}$ =FINDALLPATHS(m, n) //m,n is the length of Y^l, X

E Another Replacement Method

We propose a slightly different approach to the replacement strategy above. In brief, the approach substitutes the character in $\boldsymbol{Y^e}$ with its counterpart in $\boldsymbol{X^a}$. Subsequently, the probability $\boldsymbol{P^e}$, $\boldsymbol{P^n}$ in substitution location k are summed, and the higher resultant value is selected as the final output. The corresponding formula is presented below:

$$\boldsymbol{Y^{n'}} = [\cdots, \boldsymbol{Y^e}_{k-1}, \boldsymbol{X^a}_k, \boldsymbol{Y^e}_{k+1}, \cdots], \quad (15)$$

$$\boldsymbol{P^{n'}} = \Theta(\boldsymbol{Y^{n'}}) \quad (16)$$

$$\hat{y_k} = \begin{cases} \boldsymbol{Y^e}_k & \boldsymbol{P^{n'}}_{k,i} + \boldsymbol{P^e}_{k,i} \geq \boldsymbol{P^{n'}}_{k,j} + \boldsymbol{P^e}_{k,j}, \\ \boldsymbol{X^a}_k & \text{Otherwise.} \end{cases} \quad (17)$$

F LLMs Prompt

The prompt we use is as follows:

任务描述：请对给定的中文句子进行拼写纠错，遵循以下明确的纠错规则：1.通过替换错误的汉字来纠正句子，确保替换后的字与原字在视觉长度上保持一致，不容许删除或者增加汉字。2.在进行替换时，优先选择与原字读音或形状相似的汉字作为替换选项。3.对于句子中出现的不常见但正确的表达方式，不要进行任何修改。4.确保输出的句子中仅包含必要的文本，不加入任何额外的标点符号或解释性文字。5.如果句子没有发现任何拼写错误，直接输出原句。请根据以上规则，仅输出修改后的完整句子。

It can be translated to:

Task description: Please correct the spelling error of the given Chinese sentences, following these clear correction rules: 1.Correct the sentence by replacing the incorrect Chinese characters, ensuring that the replaced characters are visually consistent in length with the original characters, and do not delete or add any Chinese characters. 2.When making replacements, prioritize Chinese characters that are similar in pinyin or shape to the original characters as replacement options. 3.Do not modify uncommon but correct expressions in the sentence. 4.Ensure that the output sentence contains only the necessary text, without adding any additional punctuation or explanatory text. 5.If no spelling errors are found in the sentence, output the original sentence directly. Please output only the modified complete sentence according to the above rules.