C-LLM: Learn to Check Chinese Spelling Errors Character by Character

Anonymous ACL submission

Abstract

Chinese Spell Checking (CSC) aims to detect and correct spelling errors in sentences. Despite Large Language Models (LLMs) exhibit robust capabilities and are widely applied in various tasks, their performance on CSC is often unsatisfactory. We find that LLMs fail to meet the Chinese character-level constraints of the CSC task, namely equal length and phonetic similarity, leading to a performance bottleneck. Further analysis reveal that this issue stems from the granularity of tokenization, as current mixed character-word tokenization struggles to satisfy these character-level constraints. To address this issue, we propose C-LLM, a **L**arge **L**anguage **M**odel-based Chinese Spell **C**hecking method that learns to check errors **C**haracter by **C**haracter. Character-level tokenization enables the model to learn character-level alignment, effectively mitigating issues related to character-level constraints. Furthermore, CSC is simplified to replication-dominated and substitution-supplemented tasks. Experiments on two CSC benchmarks demonstrate that C-LLM achieves a 2.1% enhancement in general scenarios and a significant 12% improvement in vertical domain scenarios compared to existing methods, establishing state-of-the-art performance.

1 Introduction

Chinese Spell Checking (CSC) involves detecting and correcting erroneous characters in Chinese sentences, playing a vital role in applications (Gao et al., 2010; Yu and Li, 2014). Although Large Language Models (LLMs) exhibit potent capabilities and are increasingly being applied to a variety of tasks (Wang et al., 2023; He and Garner, 2023; Wu et al., 2023a), previous studies (Li and Shi, 2021) showed that generative models, such as LLM (Li et al., 2023a), do not perform well on CSC.

The CSC task inherently involves character-level length and phonetic constraints. The character-level length constraint requires that the predicted sentence maintain the same number of characters as the source sentence. Additionally, the phonetic constraint necessitates that the predicted characters closely match the phonetics of the source characters, as approximately 83% of spelling errors are phonetically identical or similar to the correct ones (Liu et al., 2010). We find that LLMs often fail to meet these character-level length and phonetic constraints in the CSC task.

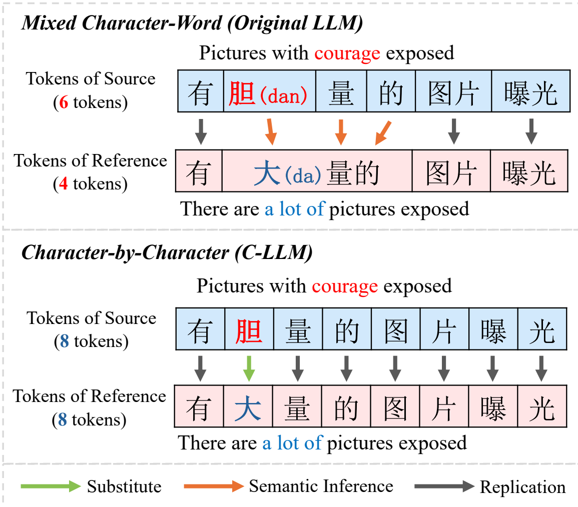

Figure 1: Encoding differences between the original LLMs and C-LLM.

Using GPT-4 (Achiam et al., 2023) as an example, we observed that under few-shot prompting, 10% of the model's predicted sentences did not match the character count of the source sentences. In contrast, this issue was entirely absent in BERT-style models. Additionally, 35% of predicted characters were phonetically dissimilar to the source characters, and errors due to non-homophone predictions account for approximately 70% of all prediction errors. These deficiencies in character length and phonetic similarity result in outputs that fail to meet task requirements, leading to suboptimal correction performance.

We find that the underlying issues lies in the granularity of the LLM's tokenization. The current mixed character-word tokenization results in a character-to-word mapping. This prevents LLMs from learning character-level alignment and tends to produce predictions that do not satisfy character-level constraints. As shown in Figure 1, under the mixed character-word tokenization, the LLM needs to infer that multiple tokens corresponds to a single token (e.g., "胆*(bold)*","大*(large)*","的*(of)*"->"大量的*(large amount)*") and deduce implicit character alignment (e.g., "胆*(bold)*"->"大*(large)*"). These complicate the CSC, as the majority of CSC cases involve simply replicating characters. For example, the correct character "量*(amount)*" is copied directly from the source. Despite the continuous advancements in the semantic understanding capabilities of LLMs across various tasks, unclear character mappings can still lead to mis-corrections and over-corrections. Therefore, it is important to establish explicit character-level alignment.

Under the premise that the source and reference sentences are of equal character-level length, training LLMs by mapping each character to a token can significantly simplify the task. Building on this concept, we propose C-LLM, a **L**arge **L**anguage **M**odel-based Chinese Spell **C**hecking method that learns to check errors **C**haracter by **C**haracter. Our motivation is to encode at the character level and establish character-level alignment for training sentence pairs, thereby alleviating the issues related to character-level constraints. As illustrated in Figure1, this approach ensures that the number of tokens in sentence pairs remains consistent, making it easier for LLMs to learn the phonetic mappings between Chinese characters. Furthermore, CSC is simplified to the tasks of replicating correct characters and replacing incorrect ones, without the need for complex reasoning.

Specifically, we construct the character-level tokenization for LLMs to ensure that tokens are encoded according to individual Chinese characters. To adapt the model to the new vocabulary, we also perform continued training on a large dataset. Furthermore, to enable the LLMs to learn CSC, we conduct supervised fine-tuning on the CSC datasets. Experiments on the general dataset CSCD-NS (Hu et al., 2022) and the multi-domain dataset LEMON (Wu et al., 2023b) show that C-LLM achieves an improvement of approximately 2.1% on the general and a significant 12% increase on the vertical domain, achieving state-of-the-art performance.

The contributions of this work can be summarized in three aspects: (1) We analyze the performance of LLM in error correction and find that mixed character-word tokenization hinders LLM from effectively understanding the character-level constraints in CSC. (2) We propose the C-LLM, which learns character-to-character alignment and can check errors character by character. (3) Through testing on general and multi-domain datasets, we found that C-LLM achieves state-of-the-art performance, providing insights for the design of future error correction models.

2 Related Work

BERT-style CSC Models With the emergence of pre-trained language models, the dominant method for CSC has shifted to BERT-style models (Devlin et al., 2019), which treat CSC as a sequence labeling task. These models map each character in a sentence to its correct counterpart and are fine-tuned on pairs of sourece and reference sentences. Additionally, some studies have integrated phonological and morphological knowledge to improve the labeling process (Cheng et al., 2020; Guo et al., 2021; Huang et al., 2021; Zhang et al., 2021). However, due to parameter constraints, these models underperform in low-frequency and complex semantic scenarios compared to LLMs.

Autoregressive CSC models Unlike BERT-style models, which can infer each token in parallel, autoregressive CSC models process tokens sequentially. Previous research (Li and Shi, 2021) indicates that autoregressive models like GPT-2 (Radford et al., 2019) may underperform on CSC. With the advancement of LLMs, several studies have investigated their text correction capabilities. The study (Li et al., 2023b) found that while ChatGPT [1] can identify the pinyin of Chinese characters, it struggles with pronunciation, making phonetic error correction challenging. Other studies (Fang et al., 2023; Wu et al., 2023a) noted that ChatGPT often produces very fluent corrections but also introduces more over-corrections. These findings align with our observations, underscoring the need to enhance LLMs' performance on CSC tasks.

3 Motivation

3.1 Problem Formulation

The CSC task aims to detect and correct all erroneous characters in Chinese sentence. Consider

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

Model	Sentence Level						Character Level					
	Detection			Correction			Detection			Correction		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
ChatGPT	63.29	49.94	55.83	58.34	46.03	51.46	64.08	53.64	58.40	57.61	48.22	52.50
GPT-4	58.50	60.23	59.35	53.35	54.93	54.13	58.52	**65.78**	61.94	51.41	57.79	54.41
BERT	78.55	**62.48**	69.60	69.16	55.02	61.28	82.76	63.41	71.80	72.02	55.18	62.49
SMBERT	**81.46**	62.40	**70.67**	73.58	56.36	63.83	**85.40**	62.70	**72.31**	76.72	56.33	64.96
SCOPE	80.75	61.57	69.87	**77.11**	**58.79**	**66.72**	84.17	62.03	71.42	**79.98**	**58.94**	**67.87**

Table 1: The performance of GPT-4 and BERT-style models (Devlin et al., 2019; Zhang et al., 2020; Li et al., 2022) on the CSCD-NS test set is evaluated at both the sentence and character levels, with precision (P), recall (R), and F1 score (F1) reported (%) for both detection (D) and correction (C) tasks.

a source sentence $X_c = \{x_1, x_2, .., x_n\}$ consisting of n characters, which may contain spelling errors. The corresponding reference sentence $Y_c = \{y_1, y_2, .., y_n\}$ contains the same number of characters as X_c, and with all errors corrected. Notably, a significant proportion of the corrected characters y_i are phonetically identical or similar to erroneous character x_i. The CSC model identifies character-level spelling mistakes in the input X_c and generates the predicted sentence $Y_c' = \{y_1', y_2', .., y_m'\}$, where y_i' is the character predicted for x_i and m should be equal to n according to the CSC. In this process, the tokens of the source sentence and the reference sentence after tokenization can be represented as $X_t = \{x_{t_1}, x_{t_2}, ..., x_{t_n}\}$ and $Y_t = \{y_{t_1}, y_{t_2}, ..., y_{t_m}\}$, respectively.

3.2 Analysis of LLMs in CSC

LLMs now exhibit powerful language processing capabilities and are widely used (Zhao et al., 2023). Similar to previous studies (Wang et al., 2023; Wu et al., 2023a), we conduct a preliminary analysis of LLM performance on the CSC using GPT-4 (Achiam et al., 2023) with in-context learning (Brown et al., 2020). Our experiments leverage the GPT-4 API and employ few-shot prompt (see Appendix A.2) on the CSCD-NS (Hu et al., 2022) test set for spelling correction. The prompt comprised five positive and five negative examples, randomly selected from the CSCD-NS training set.

As shown in Table 1, GPT-4's performance in spelling correction is inferior to that of BERT-style models. Our analysis indicates that GPT-4 struggles to meet two key constraints of the CSC task: character-level length and phonetic similarity. This misalignment results in a significant portion of the predictions that do not meet task requirements, leading to suboptimal correction performance.

Statistics reveal that 10% of GPT-4's predicted

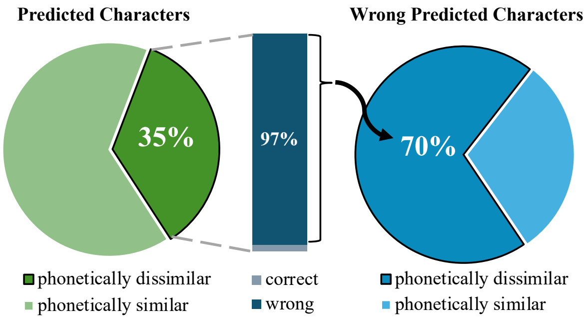

Figure 2: Statistical analysis GPT-4 from a phonetic perspective.

sentences fail to meet the character-level length constraint, adversely affecting both precision and recall. Additionally, as illustrated in Figure 2, GPT-4 generates 35% of characters that are not phonetically similar to the source ones. Among these, 97% are incorrect, and these incorrect phonologically dissimilar characters constitute a significant portion (70%) of all prediction errors, severely impacting the model's performance. Therefore, identifying the root causes of LLMs' inability to satisfy character-level length and phonetic constraints is crucial for improving their performance.

3.3 Mixed Character-Word Tokenization

By analyzing the tokenization used by the LLMs for CSC, we found that the current mixed character-word tokenization is the primary reason why LLMs struggle to meet the character-level length and phonetics constraints. Under this tokenization, sentences with spelling errors will exhibit a character-to-word mapping. This mapping can be categorized into two main cases, represented by the following formulas, where x_e and y_e denotes the erroneous character and the corresponding reference character, respectively, "=" denotes the correspondence

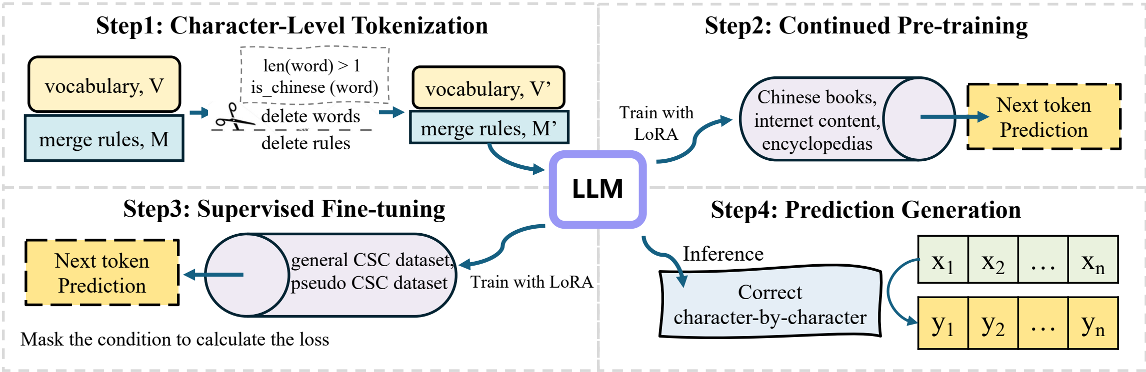

Figure 3: Overview of C-LLM. With an LLM (e.g., QWEN (Bai et al., 2023)) as the core, the implementation process of C-LLM consists of multiple steps as illustrated in the figure.

between the tokens and characters:

$$x_{t_i} = \{x_{e-1}\}, x_{t_{i+1}} = \{x_e, x_{e+1}\} \quad (1)$$
$$y_{t_i} = \{y_{e-1}, y_e, y_{e+1}\} \quad (2)$$

(1) Comparing Equation 1~ 2, the number of tokens in the source sentence does not match the reference sentence, resulting in multiple tokens corresponding to a single token.

$$x_{t_i} = \{x_{e-1}\}, x_{t_{i+1}} = \{x_e, x_{e+1}\} \quad (3)$$
$$y_{t_i} = \{y_{e-1}, y_e\}, y_{t_{i+1}} = \{y_{e+1}\} \quad (4)$$

(2) In Equation 3~ 4, even if the token counts are consistent, the characters may not align clearly due to erroneous characters and reference characters being placed in mismatched tokens.

In both cases, the mixed character-word tokenization complicates the direct alignment of x_e and y_e, necessitating inference by the model to learn the correct mappings. This transforms the CSC task into a semantic inference problem. Furthermore, inconsistencies in token counts and unclear character mappings hinder the model's ability to effectively learn character-level length and phonetic constraints.

However, in the CSC task, most correct characters in the source sentence can be directly copied during prediction, with only a small proportion of misspelled characters requiring replacement. Therefore, establishing a clear alignment between characters is crucial for this task.

4 Methodology

The CSC task requires a character-level mapping, necessitating character-by-character correction rather than token-by-token. Since current LLMs process sentences at the token level, mapping each character to a token can intuitively reduce the complexity of CSC for LLMs. Based on this concept, we propose C-LLM (as shown in Figure 3), a **L**arge **L**anguage **M**odel-based Chinese Spell **C**hecking method that learns to check errors **C**haracter by character. This approach consists of three main steps, as detailed below.

4.1 Character-Level Tokenization

The vocabulary of LLMs is typically multilingual. However, since CSC primarily addresses errors in Chinese, we only focus on the Chinese portion of the vocabulary. As shown in Equations 1~4, LLMs often map multiple characters to a single token during tokenization, complicating the CSC task by preventing a direct alignment between characters. To mitigate this issue, we construct character-level tokenization to ensure that each Chinese character is mapped to a single token. This approach facilitates a clear alignment between characters in the tokenized sentences, as represented by the following equation:

$$x_{t_i} = \{x_{e-1}\}, x_{t_{i+1}} = \{x_e\}, x_{t_{i+2}} = \{x_{e+1}\} \quad (5)$$
$$y_{t_i} = \{y_{e-1}\}, y_{t_{i+1}} = \{y_e\}, y_{t_{i+2}} = \{y_{e+1}\} \quad (6)$$

Specifically, the approach for constructing the character-level tokenization of LLM (e.g., QWEN (Bai et al., 2023)), is detailed in Algorithm 1. For the BPE (Gage, 1994) tokenization, we refine the vocabulary and the merging rules. With the new vocabulary, the model is unable to recognize words composed of multiple Chinese characters, resulting in each Chinese character being mapped to a separate token according to the revised merging rules. Experimental results indicate that the new vocabulary size is reduced to 89.2% of the original.

Models	Government	Movie	General	Game	Tech	Finance	Avg
7B-Original	8.84	50.27	12.57	37.19	28.16	10.18	24.53
7B-Char	164.12	931.99	170.02	641.76	560.99	120.99	431.65
7B-Char-PT	11.80	64.48	14.92	48.90	34.99	11.89	31.16
14B-Original	8.25	46.67	11.75	34.60	25.57	9.49	22.72
14B-Char	131.31	758.01	130.71	506.21	410.33	95.40	338.66
14B-Char-PT	10.51	58.76	14.13	44.04	32.20	11.63	28.55

Table 2: The perplexity of LLMs (e.g., QWEN1.5-14B and QWEN1.5-7B) were evaluated using the Chinese domain modeling dataset (from Skywork (Wei et al., 2023)). "Original" refers to the original LLMs, "Char" denotes LLMs with character-level tokenization, and "Char-PT" indicates the model that was further pre-trained.

Algorithm 1 Methods for Constructing Our Character-Level Tokenization.

Input:
The vocabulary of LLMs, V; The merge rules applied during tokenization, M.
Output:
The updated vocabulary V' and merge rules M' for the LLMs;

Initialization: The list of word D_w and the list of merging rules D_m to be filtered.
for $word$ in V **do**
  if len($word$) > 1 and $word$ is chinese string **then**
    add $word$ in D_w; update D_w;
  end if
end for
for $merge\_rule$ in M **do**
    $a, b = merge\_rule[0], merge\_rule[1]$
    if decode($a + b$) in D_w or decode(a) in D_w or decode(v) in D_w **then**
    add $merge\_rule$ in D_m; update D_m;
  end if
end for
Update V and M by removing the words and merge rules recorded in D_w and D_m, resulting in V' and M'.
return V' and M'.
Update the model's input and output embedding according to the new vocabulary V'.

4.2 Continued Pre-training

To mitigate the potential impact on the LLM's language modeling ability due to vocabulary constraints, we continued pre-training LLM (based on QWEN (Bai et al., 2023)) to adapt it to the new vocabulary. Specifically, we performed continued pre-training with LoRA (Hu et al., 2021) on the Chinese open-source pre-training dataset provided by Tigerbot (Chen et al., 2023b), which includes Chinese books, internet content, and encyclopedias. The training data comprised approximately 19B tokens, but we trained for 30,000 steps, covering about 2B tokens. More implementation details are provided in the Appendix A.1. The training objective was to predict the next token:

$$\mathcal{L}(\mathcal{T}) = \sum_{i=1}^{N} \log(\mathbb{P}(t_i \mid t_0, \ldots, t_{i-1}, \Theta)) \quad (7)$$

where loss is calculated as conditional probability of the i-th token t_i given the model parameters Θ.

To evaluate the impact of the character-level tokenization and continued pre-training on the LLM's language modeling ability, we measure the perplexity of LLMs using the Chinese domain modeling competency assessment dataset from Skywork (Wei et al., 2023). As shown in Table 2, the perplexity increased significantly after applying character-level tokenization, indicating a substantial impact on language modeling ability. However, this effect was mitigated after continued pre-training, bringing the language modeling ability close to that of the original LLM. This demonstrates that the model effectively adapted to the new vocabulary.

4.3 Supervised Fine-tuning

After continue pre-training, LLM only learns general language features and does not understand the specific requirements of the CSC. Therefore, supervised fine-tuning is necessary for the LLM to learn the CSC task. We utilize LoRA (Hu et al., 2021) for the fine-tuning. The training loss is defined as follows and the implementation details are provided in Appendix A.1 and Section 5.

$$\mathcal{L}(\mathcal{T}) = \sum_{i=1}^{N} \log(\mathbb{P}\left(Y_c' \mid I, X_c\right)) \quad (8)$$

where loss is calculated as the conditional probability of the predicted sentence Y_c' given the task description of the CSC I and source sentence X_c.

5 Experiments

In this section, we present the details of fine-tuning and the evaluation results of models on the two CSC benchmarks: the general dataset CSCD-NS and the multi-domain dataset LEMON.

5.1 Fine-tuning Datasets and Metrics

Datasets Previous studies (Liu et al., 2021; Xu et al., 2021) chose SIGHAN (Wu et al., 2013; Yu et al., 2014; Tseng et al., 2015) as the benchmark. However, an increasing number of studies (Hu et al., 2022; Yin and Wan, 2023; Li et al., 2022) have identified numerous issues with this dataset, such as semantically incoherent and annotation errors. Consequently, in our study, we chose two new CSC benchmarks, namely CSCD-NS and LEMON: (1) CSCD-NS (Hu et al., 2022): CSCD-NS superior in quality to SIGHAN, is the first CSC dataset where the primary source of character errors stems from pinyin input methods, containing a significant amount of homophonic and word-level errors. (2) LEMON (Wu et al., 2023b): LEMON is a novel, large-scale, multi-domain CSC dataset featuring various real-world spelling errors. It spans seven different sub-domains, typically testing the model's domain correction capabilities in a zero-shot setting. Appendix A.3 shows the data statistics.

Following the fine-tuning approach of previous work (Li et al., 2022; Liang et al., 2023), we combined the training data from CSCD-NS and 271K pseudo-data generated by ASR or OCR (denoted as Wang271K) (Wang et al., 2018) as our training set. The validation data from CSCD-NS was used as our validation set, and we test the models on the CSCD-NS test data and LEMON, respectively.

Evaluation Metrics We report sentence-level and character-level precision, recall, and F1 scores to evaluate different models. These metrics are reported separately for detection and correction tasks. We calculate metrics using the script from CSCD-NS (Hu et al., 2022). For predictions from LLMs that do not match the source sentence length, we first employ ChERRANT (Zhang et al., 2022) to extract non-equal length operations, then replace these with the source before calculating the metrics.

5.2 Baselines

We use the following CSC models for comparison. **BERT-style models**. (1) BERT (Devlin et al., 2019): BERT approaches CSC as a sequence labeling task, encoding the input sentence and employing a classifier to select the appropriate characters from the vocabulary. (2) Soft-Masked BERT (SMBERT) (Zhang et al., 2020): SMBERT composed of a detection and correction network, enhances BERT's error detection capabilities. (3) SCOPE (Li et al., 2022): SCOPE incorporates an auxiliary pronunciation prediction task with an adaptive task weighting scheme to improve CSC performance.

For the selection of LLMs, we carry out a series of experiments using QWEN1.5 (Bai et al., 2023). As one of the most potent open-source LLMs in China, QWEN exhibits robust Chinese processing capabilities and has released model parameters of multiple scales. We evaluate the performance of LLMs under the following two settings, and the prompts for LLMs are detailed in the Appendix A.2.

Fine-tuned LLM (LLM-SFT): The original LLMs (Original), the LLMs with character-level tokenization (Char), and the further pre-trained character-level LLMs (Char-PT) are each fine-tuned on the aforementioned dataset.

LLM with In-Context Learning (LLM-ICL): The original LLMs (Original), ChatGPT and GPT-4 are adapted to perform the CSC task using prompts.

5.3 Main Results

The main results on the CSCD-NS and LEMON test sets are presented in Table 3, revealing several observations: (1) Despite the robustness of both ChatGPT and GPT-4, their error correction performance is suboptimal under few-shot prompts, underscoring the critical importance of fine-tuning. (2) The LLM with character-level tokenization and without continued pre-training shows an average performance drop of approximately 1.6% compared to C-LLM (with 14B parameters). This highlights the importance of continued pre-training, which allows the model to better adapt to the new vocabulary and achieve improved performance. This is also evident from the perplexity comparison in Section 4.2. (3) C-LLM outperforms the original LLMs in error correction. For instance, the 14B parameter C-LLM shows a 1.2% improvement on general data and an average 3.7% improvement on multi-domain data, demonstrating the effectiveness of character-level correction. (4) Compared to BERT-style models, C-LLM shows superior overall performance. Specifically, the 14B parameter C-LLM surpasses the best BERT-style model, SCOPE, with an average performance improvement of 10%. It achieved a 2.1% increase in general

Models	CAR	COT	ENC	GAM	MEC	NEW	NOV	CSCD-NS	Avg
BERT (Devlin et al., 2019)	46.87	52.61	45.74	23.41	42.73	46.63	32.35	65.49	44.48
SMBERT (Zhang et al., 2020)	49.91	54.85	49.33	26.18	46.91	49.16	34.56	67.22	47.26
SCOPE (Li et al., 2022)	50.71	54.89	45.23	24.74	44.44	48.72	33.17	71.70	46.70
ChatGPT	44.88	57.11	51.46	28.78	49.85	44.40	31.77	52.50	45.09
GPT-4 (Achiam et al., 2023)	54.44	62.82	55.12	36.27	56.36	56.09	45.64	54.41	52.64
7B-Original-SFT	53.38	56.55	54.44	37.33	59.21	58.96	39.12	68.66	53.46
7B-Char-SFT	52.10	57.02	52.55	**39.00**	59.85	59.01	40.34	70.41	53.78
7B-Char-PT-SFT (C-LLM)	53.87	58.04	54.57	37.43	61.16	60.07	41.42	71.64	54.77
14B-Original-SFT	54.56	56.82	53.44	32.59	58.89	63.32	40.58	72.63	54.10
14B-Char-SFT	55.36	59.11	54.30	37.21	60.43	**65.28**	42.33	72.78	55.85
14B-Char-PT-SFT (C-LLM)	**57.54**	**60.40**	**56.48**	38.02	**65.31**	64.49	**43.92**	**73.80**	**57.49**

Table 3: Overall results of C-LLM and baseline models, are presented as character-level correction F1 scores. The best results are highlighted in bold. All the results of the BERT-style models are reproduced by us.

data and a significant 12% improvement in multi-domain data, reaching state-of-the-art results. This highlights the enhanced contextual understanding of C-LLM, particularly in vertical domain.

6 Analysis and Discussion

6.1 Scaling Trends

To further investigate the impact of model size on correction performance for LLMs, we also conduct experiments under 4B, 1.8B, and 0.5B parameters, while keeping the fine-tuning dataset and training hyperparameters consistent. As shown in Figure 4, the correction performance of the LLMs decreases on both test sets as the parameter size reduces.

Comparing C-LLM with BERT-style models, C-LLM outperforms BERT-style models at both 14B and 7B parameter sizes on the CSCD-NS and LEMON, particularly excelling in vertical domain tasks. However, smaller models exhibit weaker performance. We believe that despite the simplification of the CSC through character-level tokenization, smaller models still struggle to understand the task adequately, resulting in poor performance.

Comparing C-LLM with the original LLM, C-LLM consistently outperforms the original LLM across various parameter sizes on the CSCD dataset, although the performance gap narrows at 1.8B. This indicates that C-LLM has superior error correction capabilities compared to the original LLM. However, on the LEMON dataset, C-LLM underperforms the original LLM at sizes of 4B and smaller. We attribute this to the substantial amount of domain-specific data included in the pre-training of original LLM (Bai et al., 2023), whereas our continued pre-training for C-LLM only includes general Chinese data. This may lead to the forgetting of some domain knowledge in LLM. Larger C-LLM models (14B and 7B) suffer less from this forgetting due to their larger parameter sizes. Despite some domain knowledge being forgotten, the character-level correction approach allows larger C-LLM models to achieve better performance, while smaller models are more affected by knowledge forgetting, resulting in poorer performance.

6.2 Analysis of Length and Phonetic

Perspective	C-LLM	Original-SFT	Original-ICL
Char-to-token	98.19%	56.48%	/
Token-level	98.84%	80.54%	/
Character-level	99.78%	96.92%	22.86%

Table 4: Statistical results from the length perspective.

C-LLM alleviates issues related to character-level length constraints. To evaluate whether C-LLM effectively addresses the issue of LLMs failing to meet character-level length constraints, we analyzed from following perspectives, with results presented in Table 4.

(1) Token-Level: Our analysis shows that 98.19% of the tokens generated by C-LLM correspond one-to-one with Chinese characters. This results in approximately 18% more sentences where the token count of the source sentence matches that of the reference, compared to the original LLM.

(2) Character-Level: We select sentence pairs from CSCD-NS test set that exhibit a character-to-word mapping when tokenized by the original LLM. We then compare whether the model's output maintains the same character count as the source sentence. The results indicate that compared to Original-ICL, Original-SFT increases the proportion of maintaining character length to 96.9%, indi-

Figure 4: Overview of C-LLM. With an LLM (e.g., QWEN (Bai et al., 2023)) as the core, the implementation process of C-LLM consists of four steps as illustrated in the figure.

cating that fine-tuning helps LLM adhere to length constraints. Under C-LLM, the consistency in character length further improves to 99.8%.

These findings demonstrate that the one-to-one correspondence between tokens and Chinese characters enables LLMs to more easily generate sentences that meet character-level length constraints, resulting in superior performance.

C-LLM can reduce phonologically dissimilar predictions. We calculate the proportion of non-homophone characters in all predictions and the proportion of non-homophone errors in all incorrect predictions. As shown in Table 5, C-LLM produces fewer non-homophone prediction errors, and the ratio of these errors to the total prediction errors is reduced by 20% compared to the original LLM. This indicates that although C-LLM still generates a small number of non-homophone predictions, the impact of these errors on correction performance is significantly diminished.

	Original-SFT	C-LLM
Non-homophon Predict	8.63%	3.83%
Ratio of Wrong Predict	38.52%	18.43%

Table 5: Statistical Results for non-homophone predicted characters (under 14B model parameters).

Models	#Tokens	#Characters	AR	Time (s)
C-LLM	127057	128801	93.88%	2481.97
Original-SFT	83530	128676	86.50%	2028.77

Table 6: Analysis of Inference Speed. "AR" indicates the acceptance rate generated by draft model.

6.3 Inference Speed Analysis

Using a character-level tokenizer can decrease the model's inference speed. In this study, we perform a quantitative analysis of this impact by employing speculative decoding (Chen et al., 2023a). Our evaluation uses samples containing spelling errors from the CSCD-NS test set. The target model has 7B parameters, while the draft model has 1.8B parameters, with draft tokens set to 4. As shown in Table 6, under C-LLM, the number of decoded tokens increased by 52% compared to the original LLM, but the overall time consumption only increased by 22.33%. This is because the task complexity was reduced by C-LLM, leading to a higher acceptance rate for speculative decoding compared to the original LLM.

7 Conclusion

This paper indicates that LLMs fail to meet the Chinese character-level constraints of the CSC task, namely equal length and phonetic similarity, which hinders their correction performance. We find that the root cause lies in the granularity of tokenization, which mixes characters and words, making it difficult to satisfy these character-level constraints. To address this issue, we propose C-LLM, which establishes mappings between Chinese characters, enabling the model to learn correction relationships and phonetic similarities. This approach simplifies the CSC task to character replication and substitution. Experimental results demonstrate that C-LLM outperforms previous methods on both general and multi-domain benchmarks, achieving state-of-the-art performance.

8 Limitations

Our work has three main limitations. First, our method is specifically designed for Chinese spelling checking and may not effectively address sentences with English errors, as we did not process English words in the vocabulary. Second, our model has room for improvement, especially in handling new and trending words, which may require integrating methods such as RAG. Finally, our model's inference time is longer compared to the original model, indicating a need for further optimization for practical applications.

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*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023a. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Ye Chen, Wei Cai, Liangmin Wu, Xiaowei Li, Zhanxuan Xin, and Cong Fu. 2023b. Tigerbot: An open multilingual multitask llm. *arXiv preprint arXiv:2312.08688*.

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

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

Tao Fang, Shu Yang, Kaixin Lan, Derek F. Wong, Jinpeng Hu, Lidia S. Chao, and Yue Zhang. 2023. Is chatgpt a highly fluent grammatical error correction system? A comprehensive evaluation. *CoRR*, abs/2304.01746.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

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*.

Zhao Guo, Yuan Ni, Keqiang Wang, Wei Zhu, and Guotong Xie. 2021. Global attention decoder for chinese spelling error correction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1419–1428.

Mutian He and Philip N Garner. 2023. Can chatgpt detect intent? evaluating large language models for spoken language understanding. *arXiv preprint arXiv:2305.13512*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Yong Hu, Fandong Meng, and Jie Zhou. 2022. Cscd-ime: correcting spelling errors generated by pinyin ime. *arXiv preprint arXiv:2211.08788*.

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

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

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

Piji Li and Shuming Shi. 2021. Tail-to-tail non-autoregressive sequence prediction for chinese grammatical error correction. *arXiv preprint arXiv:2106.01609*.

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

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

Zihong Liang, Xiaojun Quan, and Qifan Wang. 2023. Disentangled phonetic representation for chinese spelling correction. *arXiv preprint arXiv:2305.14783*.

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

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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.

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

Jiaan Wang, Yunlong Liang, Fandong Meng, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. 2023. Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, et al. 2023. Skywork: A more open bilingual foundation model. *arXiv preprint arXiv:2310.19341*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Haoran Wu, Wenxuan Wang, Yuxuan Wan, Wenxiang Jiao, and Michael R. Lyu. 2023a. Chatgpt or grammarly? evaluating chatgpt on grammatical error correction benchmark. *CoRR*, abs/2303.13648.

Hongqiu Wu, Shaohua Zhang, Yuchen Zhang, and Hai Zhao. 2023b. Rethinking masked language modeling for chinese spelling correction. *arXiv preprint arXiv:2305.17721*.

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

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

Xunjian Yin and Xiaojun Wan. 2023. A comprehensive evaluation and analysis study for chinese spelling check. *arXiv preprint arXiv:2307.13655*.

Junjie Yu and Zhenghua Li. 2014. Chinese spelling error detection and correction based on language model, pronunciation, and shape. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 220–223.

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

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

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

Yue Zhang, Zhenghua Li, Zuyi Bao, Jiacheng Li, Bo Zhang, Chen Li, Fei Huang, and Min Zhang. 2022. 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, Seattle, United States. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

A Appendix

A.1 Implementation Details

Hyparameters of Continued Pre-training We provide a overview of the hyperparameter settings used in continued pre-training with LoRA (Hu et al., 2021), as illustrated in Table 7. Our implementation is based on Huggingface's Transformers (Wolf et al., 2020) in PyTorch.

Configuration	value
Learing_rate	1e-5
Adam_beta1	0.9
Adam_beta2	0.999
Adam_epsilon	1e-8
tokens/batch	2^{16}
steps	30000
lora_r	16
lora_alpha	32
lora_dropout	0.1

Table 7: Hyparameters used in continued pre-training.

Hyparameters of Supervised Fine-tuning We also provide the overview of the hyperparameter settings used in fine-tuning with LoRA (Hu et al., 2021), as illustrated in Table 8.

Configuration	value
Learing_rate	1e-4
Adam_beta1	0.9
Adam_beta2	0.999
Adam_epsilon	1e-8
num_train_epochs	10
lora_r	16
lora_alpha	32
lora_dropout	0.1

Table 8: Hyparameters used in fine-tuning.

A.2 Prompts Setting

Table 9 presents the prompts used to evaluate the error correction performance of the fine-tuned LLM, along with the few-shot prompts for ChatGPT, GPT-4 and Original-ICL. The few-shot prompt consists of 10 examples: 5 sentence pairs without typos and 5 with typos. These positive and negative examples are randomly selected from CSCD-NS, and their positions within the prompt are also randomized.

A.3 Data Statistics

The statistical results for the Wang271K, CSCD-NS and LEMON datasets are presented in Table 10. The LEMON spans seven different sub-domains, including game (GAM), encyclopedia (ENC), contract (COT), medical care (MEC), car (CAR), novel (NOV), and news (NEW). To better evaluate model performance, we filtered out sentences from the LEMON dataset where the source and reference sentences had unequal character-level lengths or where the source sentence exceeded 1000 characters.

A.4 Case Study

Table 11 compares the performance of C-LLM and the original LLM in handling character-to-word mappings. In the first example, the original LLM should map the characters *"详(comprehensive)"* and *"析(analyze)"* to the word *"详细(detail)"*. However, it incorrectly maps *"详(comprehensive)"* to *"实(accurate)"*, with the predicted characters not being phonetically similar to the source ones.

In the second example, although the correct mapping is from *"这也(as well)"* to *"这一(this)"*, the model fails to understand the relationship between the incorrect characters. It splits *"这也(as well)"* into two tokens and predicts characters that do not meet phonetic constraints. These errors indicate that the original LLM lacks a clear understanding of characters and words, making it unable to accurately correct misspelled words. In contrast, C-LLM can correctly correct misspelled characters within words through character-level tokenization. However, the third case shows that C-LLM may also make errors when correcting single incorrect characters, indicating that there is still room for improvement in our model. For some new popular words it may be necessary to combine the RAG (Lewis et al., 2020) method to do error correction.

Models	Prompts
Fine-tuned LLM	任务: 纠错文本, 输入: "原句", 输出: (Task: Correct the text, Input: $\{source\_sentence\}$, $Output$:)
ChatGPT, GPT-4 and Original-ICL	纠正句子中的错别字，并返回纠正后的句子。(Identify and correct the spelling errors in the sentence, then provide the corrected version.) $\{sentence1\}$ => $\{reference\_sentence1\}$ ... $\{sentence10\}$ => $\{reference\_sentence10\}$ => $\{source\_sentence\}$ =>

Table 9: Prompts used for testing.

Train	#Sent	#Errors	#Phonetically Similar Errors	Avg.Length
CSCD-NS	29,999	15,142	14,804	57.39
Wang271K	301,328	397,104	172,711	44.03
Dev	#Sent	#Errors	#Phonetically Similar Errors	Avg.Length
CSCD-NS	5,000	2,554	2,497	57.45
Test	#Sent	#Errors	#Phonetically Similar Errors	Avg.Length
CSCD-NS	5,000	2,528	2,484	57.63
CAR	3245	1,911	1,500	43.44
COT	993	486	341	40.11
ENC	3271	1,787	1,401	38.30
GAM	393	164	130	32.81
MEC	1942	1,032	827	39.18
NEW	5887	3,260	2,698	25.15
NOV	6000	3,415	2,585	36.24

Table 10: Statistics of the training, development and test datasets.

Models	Cases in CSCD-NS test set
Original	Src: 可/ 查询/ 详/ 析 / 数据/ 信息 Can query analyzied data information Ref: 可/ 查询/ 详细/ 数据/ 信息 Can query detailed data information Pre: 可/ 查询/ 详/ 实 / 数据/ 信息 Can query accurate data information
C-LLM	Src: 可/ 查/ 询/ 详/ 析 / 数/ 据/ 信/ 息 Can query analyzied data information Ref: 可/ 查/ 询/ 详/ 细 / 数/ 据/ 信/ 息 Can query detailed data information Pre: 可/ 查/ 询/ 详/ 细 / 数/ 据/ 信/ 息 Can query detailed data information
Original	Src: 这也 / 更新/ ，/ 让... This also update allows ... Ref: 这一 / 更新/ ，/ 让... This update allows ... Pre: 这/ 此 / 更新/ ，/ 让... This this update allows ...
C-LLM	Src: 这/ 也 / 更/ 新/ ，/ 让... This also update allows ... Ref: 这/ 一 / 更新/ ，/ 让... This update allows ... Pre: 这/ 一 / 更/ 新/ ，/ 让... This update allows ...
Original	Src: 关注/ 微信/ 火 / 下载/ 都有/ 机会 Follow WeChat fire download for a chance Ref: 关注/ 微信/ 或 / 下载/ 都有/ 机会 Follow WeChat or download for a chance Pre: 关注/ 微信/ 或 / 下载/ 都有/ 机会 Follow WeChat or download for a chance
C-LLM	Src: 关/ 注/ 微/ 信/ 火 / 下/ 载/ 都/ 有/ 机/ 会 Follow WeChat fire download for a chance Ref: 关/ 注/ 微/ 信/ 或 / 下/ 载/ 都/ 有/ 机/ 会 Follow WeChat or download for a chance Pre: 关/ 注/ 微/ 信/ 号 / 下/ 载/ 都/ 有/ 机/ 会 Follow WeChat account download for a chance

Table 11: Case study of correction results between models C-LLM and Original LLM (with 14B parameters) on the CSCD-NS test set. We mark the wrong/correct characters.