A Chunk-based Chain of Thought Prompting Method for Mitigating Over-Correction in Chinese Grammatical Error Correction

Xinquan Chang[1,2], Junguo Zhu[1,2]†
1. Faculty of Information Engineering and Automation,
Kunming University of Science and Technology, Kunming, Yunnan, 650500, China
2. Yunnan Key Laboratory of Artificial Intelligence,
Kunming University of Science and Technology, Kunming, Yunnan, 650500, China
xq.chang.kust@qq.com, jg.zhu.hit@qq.com

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in semantic understanding and text generation. However, when applied to downstream tasks such as Chinese Grammatical Error Correction (CGEC), they often suffer from over-correction issues, where grammatically correct parts are mistakenly altered. Moreover, some existing methods aim to address over-correction in Sequence-to-Sequence (Seq2Seq) models, they are difficult to adapt to decoder-only LLMs. To address these challenges, we propose a **Chunk-based Chain of Thought (CoT) Prompting Method**. Our study is structured into three key components. Initially, we identify specific types of grammatical errors in the input sentences. Following this, sentences are segmented into smaller chunks, and each chunk is analyzed to match the detected error types. Ultimately, the aggregated information guides LLMs in performing localized correction within the input sentences. The experimental results have proved the effectiveness of our method in mitigating over-correction, achieving higher $F_{0.5}$ score while maintaining robust grammatical error correction performance. This method provides innovative perspectives on employing LLMs to enhance the precision and granularity of CGEC task.

Keywords: Chinese Grammatical Error Correction , Over-correction , Chain of Thought , Large Language Model

1 Introduction

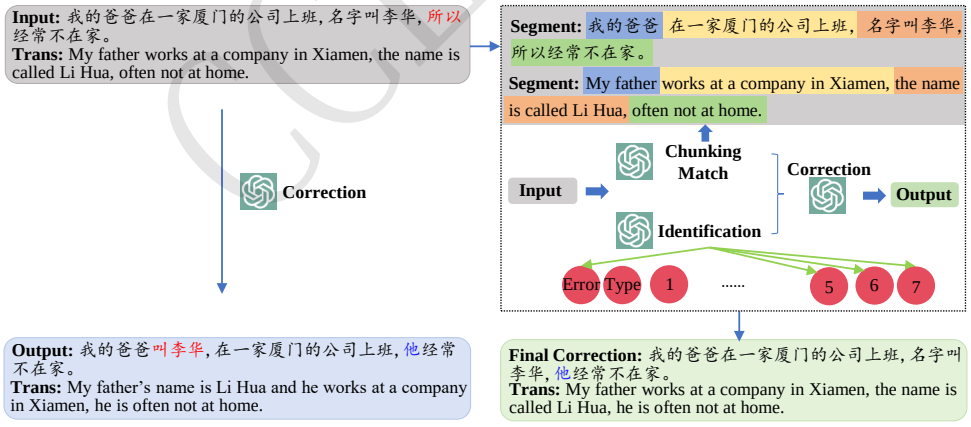

Figure 1: Comparison diagram of direct correction and our method. The right part is an illustration of addressing over-correction through chunk-based CoT prompting method. Grammatical errors and over-correction characters are highlighted in red, correct modifications are highlighted in blue. Trans stands for respective translations.

† Corresponding author

Chinese Grammatical Error Correction (CGEC) focuses on identifying and correcting grammatical errors in Chinese sentences, such as punctuation, spelling, and grammatical issues (Zhao et al., 2018), adhering to the minimal-editing principle to produce a corrected version (Zhang et al., 2022a). CGEC is crucial for improving translation systems, automatic proofreading tools, and language teaching (Wang et al., 2021). The primary frameworks of CGEC include Sequence-to-Sequence (Seq2Seq) and Sequence-to-Edit (Seq2Edit). The Seq2Seq framework treats CGEC as a monolingual translation task (Yuan and Briscoe, 2016; Lewis et al., 2020), while the Seq2Edit framework transforms the input sentences into a series of editing operations (Stahlberg and Kumar, 2020). Although Seq2Seq offers flexibility, it is often susceptible to over-correction, a phenomenon where models modify parts of the sentence that were originally correct while correcting grammatical errors. In contrast, Seq2Edit provides greater precision but necessitates the design of complex editing operations. Recently, advanced approaches have integrated two frameworks, aiming to combine their respective advantages (Li et al., 2023a). The state-of-the-art (SOTA) methods leverage the strengths of both Seq2Seq and Seq2Edit to enhance correction performance.

Traditional methods based on Seq2Seq and Seq2Edit often rely on expensive labeled data for pre-training. In contrast, decoder-only Large Language Models (LLMs) can be pre-trained on large-scale unlabeled data, enabling them to achieve exceptional performance in CGEC task (Fan et al., 2023). Existing research has demonstrated that when LLMs are applied to CGEC, they exhibit notable corrective capabilities, generating revised sentences with high fluency (Li et al., 2024). However, their generation mechanism, which prioritizes high-probability expressions by substituting low-frequency words with more frequent alternatives, often leads to deviations from the minimal-editing principle (Qu and Wu, 2023). Specifically, this probabilistic bias causes LLMs to not only correct erroneous segments but also unnecessarily modify correct portions of the input, resulting in over-correction. As a result, their performance in CGEC falls short when compared to low-parameter, optimal models specifically designed for CGEC (Li et al., 2023b). The left part of Figure 1 illustrates an example of over-correction issues generated by LLMs. This direct correction method may over-correct or change the intended meaning, violating the minimal-editing principles. Moreover, existing methods aim to address over-correction in Seq2Seq models, they are often difficult to adapt to decoder-only LLMs.

To address the challenges described above, we propose a method based on chunk-based Chain of Thought (CoT) prompting method. CoT is a reasoning process that involves breaking down complex problems into a sequence of logical steps. The right part of figure 1 shows our Chunk-based CoT prompting method. Our method introduces a structured approach that first identifies grammatical error types in the input sentences. It then segments these sentences into smaller, more manageable chunks, and matches each chunk with known error categories. Finally, the model integrates the insights from the chunk-level analysis to generate corrections, ensuring both grammatical accuracy and fidelity to the original sentence. This allows LLMs to focus on localized errors and mitigate over-correction. Our contributions can be summarized as follows:

• To mitigate the propensity of LLMs for over-correction when performing CGEC task, we propose a chunk-based CoT prompting method. Experimental results on three datasets validate the effectiveness and robustness of our method. Compared with direct correction, our method achieves higher $F_{0.5}$ score and mitigate over-correction problem. In addition, we find that LLMs perform worse in CGEC than the traditional optimal model based on Seq2Seq or Seq2Edit, which is also worthy of further exploration.

• Our proposed prompting templates offer a novel insight for conducting CGEC, and our method is training-free and can be applied to general-purpose LLMs. We also analyze the influence of our method on different error types in detail. And the thorough analysis of different chunk granularity demonstrates the role of chunking mechanism in performance improvement.

2 Related Work

2.1 Sequence-to-Sequence CGEC

CGEC has garnered significant research interest in recent years due to its importance in improving the quality of written communication (Zhao et al., 2019). Early methods primarily focused on Seq2Seq mod-

Proceedings of the 24th China National Conference on Computational Linguistics, pages 831-841, Jinan, China, August 11-14, 2025.

els, which utilize encoder-decoder architectures to transform erroneous input sequences into corrected output sequences (Lewis et al., 2020; Zhao and Wang, 2020). These models typically rely on large-scale datasets to learn grammatical patterns and error correction strategies. Existing optimal models also supplement additional information on the basis of seq2seq (Zhang et al., 2022b). However, despite their success, they are prone to issues such as over-correction.

2.2 Sequence-to-Edit CGEC

To address over-correction, Seq2Edit models were introduced as an improvement (Omelianchuk et al., 2020). These models don't generate entirely new sequences but rather predict a series of edits (insertions, deletions, and substitutions) on the input sequence (Liang et al., 2020; Zhang et al., 2022a). This approach is more targeted and reduces the risk of over-correction by focusing on minimal, precise adjustments. The development of such models has led to improved accuracy in CGEC, especially in handling complex grammatical errors in Chinese. Seq2Edit models require the design of complex editing operations, which often result in a compromise in the fluency of the generated sentences. In contrast, our approach emphasizes localized error correction by segmenting sentences into smaller, more manageable chunks. Each chunk is analyzed in relation to specific error types. The LLMs use the aggregated information as a guide to generate fluent sentences. This method allows for precise error localization while preserving high fluency in the corrected output.

2.3 Detection-Correction CGEC

Recent approaches to CGEC typically divide the task into two stages: error detection and correction. SOTA methods combine the strengths of both Seq2Seq and Seq2Edit models to enhance correction performance. Li et al. (Li et al., 2023a) utilize detection labels from the Seq2Edit model to construct templates, which are then used in conjunction with a Seq2Seq model for consistency learning, thereby improving correction accuracy. Similarly, DeCoGLM (Li and Wang, 2024)integrates both detection and correction into a unified language model, fostering the multi-task learning capability of a single model to enhance the overall correction effectiveness. Recent works have explored the use of CoT prompting to enhance the performance of LLMs in generation and reasoning tasks (Lu et al., 2024). Leveraging the strengths of prior approaches, we propose a CoT prompting method that executes grammatical error correction in multiple steps. By identifying grammatical error types and performing sentence segmentation, our approach achieves more precise error localization. This method effectively mitigates the over-correction issue and enhances the overall accuracy in LLMs. Moreover, our method exhibits remarkable universality. It can be applied to the general LLMs in a training-free way without additional fine-tuning or training of the model.

3 Methodology

As depicted in Figure 2, we developed the Chunk-based CoT prompting method by surveying existing prompt templates (Fang et al., 2023; Qu and Wu, 2023) and integrating our grammatical error correction strategy. The process consists of three main stages: Grammatical Error Type Identification, Sentence Chunking and Error Matching, and Chunk-Level Correction. Accordingly, we broadly divide the model fine-tuning process into two stages. In stage 1, we fine-tune a classification model on the baseline to identify the types of grammatical errors. Following this, we fine-tune the generation model to complete CGEC in stage 2. In this section, we introduce the Chunk-based CoT prompting method proposed for CGEC.

3.1 Grammatical Error Type Identification

The first step is to identify the types of grammatical errors present in input sentences. Consistent with prior work on FCGEC (Xu et al., 2022), we define seven error types, as shown in Table 1, which include: Incorrect Word Order (IWO), Incorrect Word Collocation (IWC), Component Missing (CM), Component Redundancy (CR), Structure Confusion (SC), Illogical (ILL), and Ambiguity (AM). These error types provide a clear framework for the model to identify and correct specific grammatical issues.

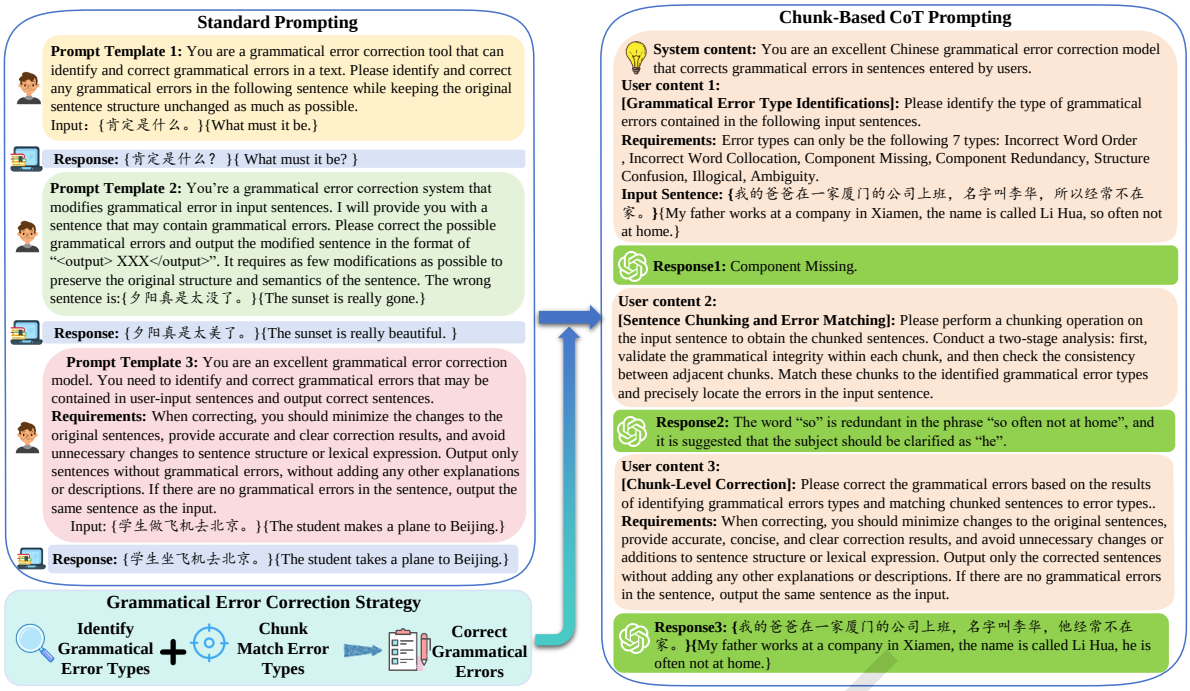

Figure 2: An illustration of standard prompting and our chunk-based CoT prompting.

In Stage 1, the focus is on optimizing the classification task. We employ the binary cross-entropy loss function and save the model parameters upon completion of training.

$$\hat{y}_i = \sigma(z_i) = \frac{1}{1 + e^{-z_i}} \tag{1}$$

where z_i is the original output of the i_{th} error type, $\hat{y}_i$ is the predicted probability, and σ is the Sigmoid function.

$$\mathcal{L}_{\text{classify}} = -\sum_{i=1}^{C} [y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)] \tag{2}$$

where C is the number of error types, and y_i is the label of the class i error. Specifically, we use the LoRA technique in LLaMA-Factory (Zheng et al., 2024) to accomplish supervised fine-tuning.

3.2 Sentence Chunking and Error Matching

Building upon the error type identification, we implement a chunking mechanism. To ensure the coherence of the step-by-step reasoning in LLMs, we employ a segmentation driven by LLMs themselves to decompose the input sentences into syntactically coherent units. Each chunk is analyzed through two stages: internal validation to ensure grammatical correctness within local contexts, and external consistency checking to evaluate the logical coherence between adjacent segments. These help reduce the complexity of syntactic dependencies and enables the model to more effectively address localized error patterns.

After chunking, each segment is matched with its corresponding error type. This pairing allows the model to address specific errors by limiting the scope of modifications, thereby minimizing the risk of over-correction.

3.3 Chunk-Level Correction

In Stage 2, the emphasis shifts to optimizing the grammatical error correction task. Here, we utilize the cross-entropy loss function for the generation task. To maintain the integrity of the classification task,

Type	Example
IWO	全场职工讨论并听取了意见。 The entire staff discussed and listened to the comments. Tips: listen to the comments first and discuss it later.
IWC	我达到了很大的进步。 I reached a lot of progress. Tips: ”progress” should be combined with ”make”.
CM	常说劳逸结合。 Often say work and rest combine. Tips: Lack of subject ”The people”.
CR	时间大约过去了20分钟左右。 Time has passed about 20 minutes or so. Tips: ”about” and ”or so” are redundant.
SC	交通事故发生的原因是开车看手机造成的。 Traffic accidents are caused by (because) looking at cell phones while driving. Tips: the structure of ”because” and ”caused by” cannot appear together in one sentence.
ILL	我们应该防止事故不发生。 We should prevent accidents from not occurring. Tips: double negation causes illogical errors.
AM	我们要把水留给晚上来的人。 We need to keep the water for those who come up later/at night. Tips: there is an ambiguity about when the person came.

Table 1: Examples and illustrations of different grammatical errors types.

we freeze its parameters and only train the parameters related to the generation task.

$$\mathcal{L}_{\text{generate}} = -\sum_{t=1}^{T} \log P(y_t|y < t) \tag{3}$$

where T is the length of the target text, y_t is the target word element, and $P(y_t|y < t)$ represents the probability of model generating the current word y_t given the preceding context $y < t$.

We optimize only one task at each stage to avoid conflicts between task objectives. The classification and generation tasks have distinct optimization goals, and training them separately eliminates the need for complex hyperparameter tuning that would be required in joint training.

4 Experiments

4.1 Dataset and Evaluation

For dataset, we follow the previous work. We employ a combination of the FCGEC *train* set (Xu et al., 2022), the Chinese Lang8 dataset (Zhao et al., 2018) and the HSK dataset (Zhang et al., 2022a) as our training set. We utilize MuCGEC *dev* (Zhang et al., 2022a), FCGEC *dev* and NaCGEC *dev* (Ma et al., 2022) as our development sets. The models are evaluated on NLPCC18 *test* (Zhao et al., 2018), FCGEC *test* and NaCGEC *test*. It is worth noting that we randomly selected 1000 data samples from the FCGEC *dev* to serve as the test set for subsequent experimental validation. The detailed dataset description is shown in Table 2.

We evaluate models' performance with Precision, Recall, and $F_{0.5}$ score ($F_{0.5}$) from word-level and character-level respectively. The $F_{0.5}$ metric assigns greater weight to precision, thus prioritizing the precision of corrections. We adopt the official implementation of MaxMatch (M^2) (Dahlmeier and

Dataset	Source	#Sentences	Usage
FCGEC *train*	Native	36341	Fine-tuning (Stage 1)
HSK	Learner	156870	Fine-tuning (Stage 2)
Lang8	Learner	1220960	Fine-tuning (Stage 2)
MuCGEC *dev*	Learner	2467	Validation
FCGEC *dev*	Native	1000	Validation
NaCGEC *dev*	Native	500	Validation
NaCGEC *test*	Native	5869	Testing
NLPCC18 *test*	Learner	2000	Testing
FCGEC *test*	Native	3000	Testing
FCGEC *dev*	Native	1000	Testing

Table 2: Statistics of the CGEC datasets used in this paper.

Ng, 2012) scorer to calculate word-level $F_{0.5}$ score on NLPCC18 *test* and choose PKUNLP as our word segment tool. We apply ChERRANT tool (Zhang et al., 2022a) to report character-level metric calculation for FCGEC *test* and NaCGEC *test*.

4.2 Model settings

Previous models BART is a Seq2Seq model based on Transformers architecture (Vaswani et al., 2017). GECToR is an efficient Seq2Edit system based on Transformer encoder (Omelianchuk et al., 2020). SynGEC (Zhang et al., 2022b) incorporates syntax information into Seq2Seq models. TemplateGEC (Li et al., 2023a) uses the output of the GECToR model as supplementary information for Seq2Seq models. Chunk-CoT represents our chunk-based CoT prompting method.

Resource-restricted LLMs LLaMA3-8B-Chinese (LLaMA3-8B), the latest instruction fine-tuning generation models, is chosen as the baseline model. Baichuan2-7B has surpassed most open-source models in benchmarks for language generation and coding. We choose Baichuan2-7B (Yang et al., 2023) as the baseline model.

General-Purpose LLMs We selected the GPT-4o mini model, which features a parameter size of approximately 8 billion, and the DeepSeek-V2.5 model, which has a substantial parameter count of approximately 236 billion. For evaluation, we utilized the respective API interfaces, setting the maximum token limit to 1024 and employing the generation temperature of 0.7.

4.3 Main Results

The main results are presented in Table 3. We observe that, by comparing against the baseline fine-tuning models, the Chunk-CoT method achieves an improvement in $F_{0.5}$ scores across all datasets. Notably, the application of the Chunk-CoT method significantly enhances the grammatical error correction performance of models, irrespective of whether they are resource-constrained large models or general-purpose large models. This demonstrates the effectiveness and robustness of our approach. Besides, the substantial improvement in precision across diffierent datasets indicates the efficacy of our method in mitigating the over-correction issue prevalent in LLMs. Our method results in a slight decrease in Recall, a common trade-off in CGEC task (Ng et al., 2014), as incorrect corrections are generally more detrimental than omissions. Further analysis regarding the effect of Chunk-CoT on reducing over-correction is presented in Section 5.1.

Additionally, several intriguing observations have been made. While Chunk-CoT prompting improves performance, decoder-only LLMs such as Baichuan2-7B and LLaMA3-8B still fall short of BART, primarily due to BART's specialized pre-training on denoising tasks—including token masking, deletion, and text infilling that inherently aligns with grammatical error correction. Although Chunk-based CoT underperforms traditional models on certain test sets, it effectively mitigates over-correction and enhances accuracy without requiring complex fine-tuning, thereby demonstrating practical adaptability and value for real-world CGEC applications. Secondly, the degree of improvement varies across different

Proceedings of the 24th China National Conference on Computational Linguistics, pages 831-841, Jinan, China, August 11-14, 2025.

Model	Methods	NLPCC18 *test*			FCGEC *test*			NaCGEC *test*		
		Precision	Recall	$F_{0.5}$	Precision	Recall	$F_{0.5}$	Precision	Recall	$F_{0.5}$
Previous Models										
GECToR	-	-	-	-	**46.11**	34.35	**43.16**	-	-	-
BART	-	48.80	**33.50**	44.70	38.38	**37.62**	38.23	50.95	31.29	45.26
SynGEC	-	49.96	33.04	45.32	-	-	-	**51.45**	**39.69**	**48.57**
TemplateGEC	-	**54.50**	27.40	**45.50**	-	-	-	-	-	-
Resource-restricted LLMs										
LLaMA3-8B	Fine-tuning	46.41	27.86	40.96	42.87	**30.12**	39.52	60.86	**30.14**	50.55
	+Chunk-CoT	**48.19**	**28.28**	**42.24**	**45.76**	27.73	**40.49**	**63.54**	29.72	**51.76**
Baichuan2-7B	Fine-tuning	51.43	**28.09**	44.10	43.86	**39.88**	43.00	61.72	44.00	57.12
	+Chunk-CoT	**52.17**	27.79	**44.38**	**46.21**	36.94	**44.00**	**63.46**	**44.78**	**58.57**
General-Purpose LLMs										
GPT-4o mini	Zero-shot	25.76	26.95	25.99	18.39	**14.81**	17.54	11.75	11.42	11.68
	Chunk-CoT	**26.95**	**27.42**	**27.04**	**22.62**	12.64	**19.54**	**14.43**	**11.63**	**13.77**
DeepSeek-V2.5	Zero-shot	30.23	**31.56**	30.49	21.29	**20.81**	21.19	16.43	**16.33**	16.41
	Chunk-CoT	**36.76**	29.99	**35.17**	**23.67**	19.46	**22.69**	**19.16**	15.29	**18.24**

Table 3: Main results on NLPCC18 *test*, NaCGEC *test*, and FCGEC *test*. The results of previous model experiments are cited from the original text. The best performance in each set of experiments is highlighted in bold.

models when employing Chunk-CoT. Baichuan2-7B consistently outperforms LLaMA3-8B, likely due to the distinct capabilities they acquire through pre-training. Lastly, there are variations in performance for the same model across different datasets, which may correlate with the scale, difficulty, and characteristics of the datasets. For instance, DeepSeek-V2.5 exhibits superior performance on the NLPCC18-Test dataset compared to other datasets. This may be attributed to the fact that the dataset originates from learners, a characteristic that closely aligns with the model's capabilities.

5 Analysis

5.1 Over-correction Mitigation

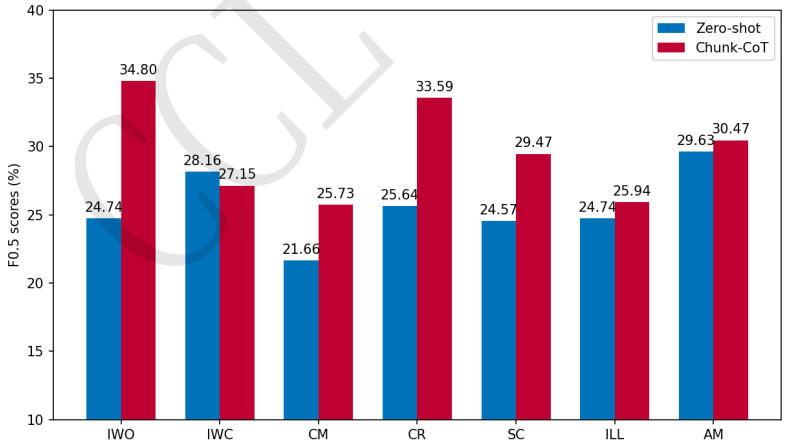

Figure 3: Comparison of $F_{0.5}$ for seven grammatical error types on the FCGEC *dev* using DeepSeek-V2.5.

To further verify the effectiveness of Chunk-CoT in mitigating over-correction, we use DeepSeek-V2.5 as the backbone and present fine-grained $F_0.5$ results across the seven categories of CGEC errors in Figure 3. We present in Table 4 the number of over-corrections and under-corrections that Chunk-CoT reduces on diffierent error types.

The results depicted in Figure 3 show that Chunk-CoT method outperforms the Zero-shot in terms of $F_{0.5}$ scores across most error types. Notably, the most significant improvements are observed in IWO

Proceedings of the 24th China National Conference on Computational Linguistics, pages 831-841, Jinan, China, August 11-14, 2025.

Type	#Over-corrections / #Under-corrections	
	Zero-shot	Chunk-CoT
IWO	76 / 107	45(-40.7%)/ 101
IWC	156 / 154	153(-1.9%)/ 165
CM	238 / 169	193(-18.9%)/ 166
CR	95 / 83	68(-28.4%)/ 74
SC	190 / 161	148(-22.1%)/ 150
ILL	128 / 142	120(-6.3%) / 148
AM	38 / 38	38(0%)/ 42
ALL	921 / 855	765(-16.9%)/ 847

Table 4: Using DeepSeek-V2.5, Chunk-CoT reduced the number of over-corrections and under-corrections for different error types on FCGEC *dev* compared to Zero-shot.

and CR. However, for IWC errors, the $F_{0.5}$ of Chunk-CoT is lower than that of Zero-shot. This may be attributed to the fact that IWC errors require a deeper understanding of context, which Chunk-CoT fails to fully leverage, resulting in reduced performance. The results in Table 4 further demonstrate that Chunk-CoT effectively reduces the number of over-correction without deteriorating under-correction. These findings support the effectiveness of Chunk-CoT in mitigating over-correction induced by LLMs and enhance its robustness across different error types.

5.2 Ablation Study

Method	NLPCC18 *test*			NaCGEC *test*		
	Precision	Recall	$F_{0.5}$	Precision	Recall	$F_{0.5}$
Baichuan2-7B						
Fine-tuning	51.43	**28.09**	44.10	61.72	44.00	57.12
w/o Chunking Mechanism	51.88	27.62	44.13	63.40	43.65	58.14
w/o CoT Prompting	51.47	27.93	44.05	62.43	44.53	57.78
Chunk-CoT(ours)	**52.17**	27.79	**44.38**	**63.46**	**44.78**	**58.57**
DeepSeek-V2.5						
Zero-shot	30.23	31.56	30.49	16.43	**16.33**	16.41
w/o Chunking Mechanism	33.24	**32.64**	33.12	17.83	15.78	17.38
w/o CoT Prompting	30.89	30.48	30.81	16.77	15.77	16.56
Chunk-CoT(ours)	**36.76**	29.99	**35.17**	**19.16**	15.29	**18.24**

Table 5: Ablation Experiment results.

To assess the contributions of key components within our proposed method, we conduct ablation studies using DeepSeek-V2.5 and Baichuan2-7B models on the NLPCC18 *test* and NaCGEC *test* datasets. We defined "w/o chunk" as the application of the identification-correction CoT approach without incorporating the chunking operation on the input sentences. Conversely, "w/o CoT" refers to the direct correction of input sentences after chunking, without employing the CoT method. The experimental results are presented in Table 5.

Our results indicate that eliminating either chunking mechanism or CoT prompting leads to a decline in correction performance. Notably, the absence of CoT has a more substantial negative impact, indicating its pivotal role in enhancing the correction capabilities of our approach. Optimal model performance is achieved when both components are integrated.

5.3 Impact of Chunking Granularity

To assess the influence of chunking granularity on CGEC performance, we conduct experiments employing different chunking strategies. We evaluated four configurations: no chunking, phrase-based

Chunk Granularity	NLPCC18 *test*			NaCGEC *test*		
	Precision	Recall	$F_{0.5}$	Precision	Recall	$F_{0.5}$
No Chunking	33.24	**32.64**	33.12	17.83	15.78	17.38
Phrase-based Chunking	31.62	29.94	31.27	15.58	14.97	15.45
Punctuation-based Chunking	34.72	31.25	33.97	18.12	**16.49**	17.77
LLMs-driven Chunking	**36.76**	29.99	**35.17**	**19.16**	15.29	**18.24**

Table 6: Effect of different chunking granularities on correction performance

chunking, punctuation-based chunking, and LLMs-driven chunking, where LLMs autonomously segment sentences. For phrase-based chunking, we implemented segmentation using the jieba tool [1] based on phrase rules.

The results presented in Table 6 indicate that the LLMs-driven chunking approach achieves the best performance across both test sets, demonstrating its ability to dynamically segment sentences based on contextual and semantic cues. This context-aware strategy effectively mitigates over-correction, a limitation observed in the no-chunking baseline, which processes entire sentences as single units and consequently exhibits average performance. In contrast, phrase-based chunking underperforms, likely due to mismatches between its rigid segmentation strategy and the diverse structures in the test data. While punctuation-based chunking shows moderate improvements, its reliance on surface-level cues limits adaptability. The LLMs-driven method accurately identifies syntactic and semantic boundaries, producing coherent chunks that enhance grammatical error correction. These findings underscore the critical role of adaptive chunking mechanisms in optimizing LLM-based CGEC systems, as they balance segmentation flexibility with grammatical integrity.

6 Conclusion

To address the over-correction issue of LLMs in CGEC, we propose a chunk-based CoT prompting method. By integrating chunking mechanism and CoT into LLMs, our method enables models to correct step-by-step, thereby achieving accurate error localization and localized correction. Experimental results across diverse datasets and LLMs demonstrate that enhanced LLMs effectively mitigate over-correction issues while achieving higher precision, thereby validating the effectiveness and robustness of our method. Detailed analyses further illustrate the effectiveness of our method across different error types and highlight the role of chunking in performance improvement. Our proposed method provides a novel insight to leveraging LLMs for fine-grained correction, which can be directly applied to general-purpose LLMs in a training-free manner.

However, our approach also has certain limitations. First, our method is language-independent and can be generalized to other languages. Second, identifying error types and performing chunking introduces additional training costs and inference time. Third, our study only explores a limited number of currently available LLMs. Although our method improves the correction performance of LLMs, it still lags behind the SOTA Seq2Seq models with fewer parameters. In future work, we will continue to explore ways to enhance the performance of LLMs in CGEC.

Acknowledgements

We sincerely appreciate the constructive feedback provided by the anonymous reviewers. We express our gratitude to the individuals and laboratories for their technical support. This research is supported by the Regional Fund Project of National Natural Science Foundation of China (62166022), and Yunnan Xingdian Talent Support Plan Project(KKXX202403023).

[1] https://github.com/fxsjy/jieba

References

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 568–572.

Yaxin Fan, Feng Jiang, Peifeng Li, and Haizhou Li. 2023. Grammargpt: Exploring open-source llms for native chinese grammatical error correction with supervised fine-tuning. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 69–80. Springer.

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. *arXiv preprint arXiv:2304.01746*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Wei Li and Houfeng Wang. 2024. Detection-correction structure via general language model for grammatical error correction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1748–1763.

Yinghao Li, Xuebo Liu, Shuo Wang, Peiyuan Gong, Derek F Wong, Yang Gao, He-Yan Huang, and Min Zhang. 2023a. Templategec: Improving grammatical error correction with detection template. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6878–6892.

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. *arXiv preprint arXiv:2307.09007*.

Yinghui Li, Shang Qin, Jingheng Ye, Shirong Ma, Yangning Li, Libo Qin, Xuming Hu, Wenhao Jiang, Hai-Tao Zheng, and Philip S Yu. 2024. Rethinking the roles of large language models in chinese grammatical error correction. *arXiv preprint arXiv:2402.11420*.

Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, and Jinpeng Dong. 2020. Bert enhanced neural machine translation and sequence tagging model for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 57–66.

Qingyu Lu, Baopu Qiu, Liang Ding, Kanjian Zhang, Tom Kocmi, and Dacheng Tao. 2024. Error analysis prompting enables human-like translation evaluation in large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 8801–8816.

Shirong Ma, Yinghui Li, Rongyi Sun, Qingyu Zhou, Shulin Huang, Ding Zhang, Li Yangning, Ruiyang Liu, Zhongli Li, Yunbo Cao, et al. 2022. Linguistic rules-based corpus generation for native chinese grammatical error correction. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 576–589.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the eighteenth conference on computational natural language learning: shared task*, pages 1–14.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: tag, not rewrite. *arXiv preprint arXiv:2005.12592*.

Fanyi Qu and Yunfang Wu. 2023. Evaluating the capability of large-scale language models on chinese grammatical error correction task. *arXiv preprint arXiv:2307.03972*.

Felix Stahlberg and Shankar Kumar. 2020. Seq2edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5147–5159.

Chang Su, Xiaofeng Zhao, Xiaosong Qiao, Min Zhang, Hao Yang, Junhao Zhu, Ming Zhu, and Wenbing Ma. 2023. Hwcgec: hw-tsc's 2023 submission for the nlpcc2023's chinese grammatical error correction task. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 59–68. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yu Wang, Yuelin Wang, Kai Dang, Jie Liu, and Zhuo Liu. 2021. A comprehensive survey of grammatical error correction. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 12(5):1–51.

Lvxiaowei Xu, Jianwang Wu, Jiawei Peng, Jiayu Fu, and Ming Cai. 2022. Fcgec: Fine-grained corpus for chinese grammatical error correction. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1900–1918.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 conference of the north American Chapter of the Association for computational linguistics: Human language technologies*, pages 380–386.

Yue Zhang, Zhenghua Li, Zuyi Bao, Jiacheng Li, Bo Zhang, Chen Li, Fei Huang, and Min Zhang. 2022a. 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.

Yue Zhang, Bo Zhang, Zhenghua Li, Zuyi Bao, Chen Li, and Min Zhang. 2022b. Syngec: Syntax-enhanced grammatical error correction with a tailored gec-oriented parser. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2518–2531.

Zewei Zhao and Houfeng Wang. 2020. Maskgec: Improving neural grammatical error correction via dynamic masking. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 1226–1233.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the nlpcc 2018 shared task: Grammatical error correction. In *Natural Language Processing and Chinese Computing: 7th CCF International Conference, NLPCC 2018, Hohhot, China, August 26–30, 2018, Proceedings, Part II 7*, pages 439–445. Springer.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North*. Association for Computational Linguistics.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410.