Chinese Grammatical Error Correction via Large Language Model Guided Optimization Training

Xiao Liu[1,2], **Ying Li**[1,2]✉, **Zhengtao Yu**[1,2]
[1]Faculty of Information Engineering and Automation,
Kunming University of Science and Technology, Kunming, China
[2]Yunnan Key Laboratory of Artificial Intelligence, Kunming, China
{sakuracedia,yingli_hlt}@foxmail.com, ztyu@hotmail.com

Abstract

Pre-trained language model-based methods for Chinese Grammatical Error Correction (CGEC) are categorized into Seq2Seq and Seq2Edit types. However, both Seq2Seq and Seq2Edit models depend on high-quality training data significantly. Considering the strong generation and inference ability of large language models (LLMs), we propose a large language model-guided optimization training method to exploit LLMs to extract error knowledge to optimize the traditional CGEC model training process. On the one hand, we use error types and confusion sets as extra knowledge to guide LLMs to generate diverse pseudo data, thus extending the error distribution of our training data. On the other hand, LLMs are utilized to infer the predicted results from our CGEC models and obtain the re-training data, thus iteratively optimizing our pre-trained CGEC models. Experiments on two benchmark datasets show that our LLMs-guided optimization method with small-scale training data can achieve comparable results with baseline models with large-scale training data. Detailed comparison experiments demonstrate that both the early deviser pseudo data and the later re-training data are extremely useful for traditional CGEC model optimization training, and can benefit from each other. We will release our code and prompts at https://github.com/SakuraAcedia/llm-cgec-got to facilitate future work.

1 Introduction

Grammatical Error Correction (GEC) aims to detect and correct any possible grammatical errors in a given sentence (Wang et al., 2021), i.e., missing, redundant, substitution, and word-order four error types (Zhang et al., 2022a). As shown in Figure 1, the traditional GEC model receives a sentence containing errors and generates its corrected one. The GEC results have widely applied to various artificial intelligence tasks such as writing scenarios, speech recognition fields, and information retrieval scenes (Bryant et al., 2022; Zhang et al., 2023; Wang et al., 2023a).

Current typical GEC methods can be categorized into two lines, i.e., Seq2Seq models and Seq2Edit models. Seq2Seq models treat GEC as a generation task, which takes an erroneous sentence as input and obtains its corrected one autoregressively (Zhang et al., 2022b; Li et al., 2023a). Seq2Edit models regard GEC as a token classification task, which first predicts a sequence of text-editing operations and then generates target tokens based on the sequence operation labels (Zhang et al., 2022a; Omelianchuk et al., 2020). With the advance in pre-trained language models, both Seq2Seq and Seq2Edit models have achieved obvious improvements in Chinese grammatical error correction. However, their performances are highly dependent on the quality and scale of the training corpus (Bryant et al., 2022).

In the past few years, different data augmentation techniques have been extensively studied (Koyama et al., 2021; Stahlberg and Kumar, 2021; Ye et al., 2023). On the one hand, researchers attempt to decline the error distributions between synthetic and reality corpus by back-translation or error pattern constraint (Choe et al., 2019; Stahlberg and Kumar, 2021). On the other hand, some works exploit noise injection or MixEdit approaches to increase the diversity of synthetic corpus (Koyama et al., 2021; Ye et

✉ Corresponding author.

Proceedings of the 23rd China National Conference on Computational Linguistics, pages 1366-1380, Taiyuan, China, July 25 - 28, 2024.
Volume 1: Main Conference Papers

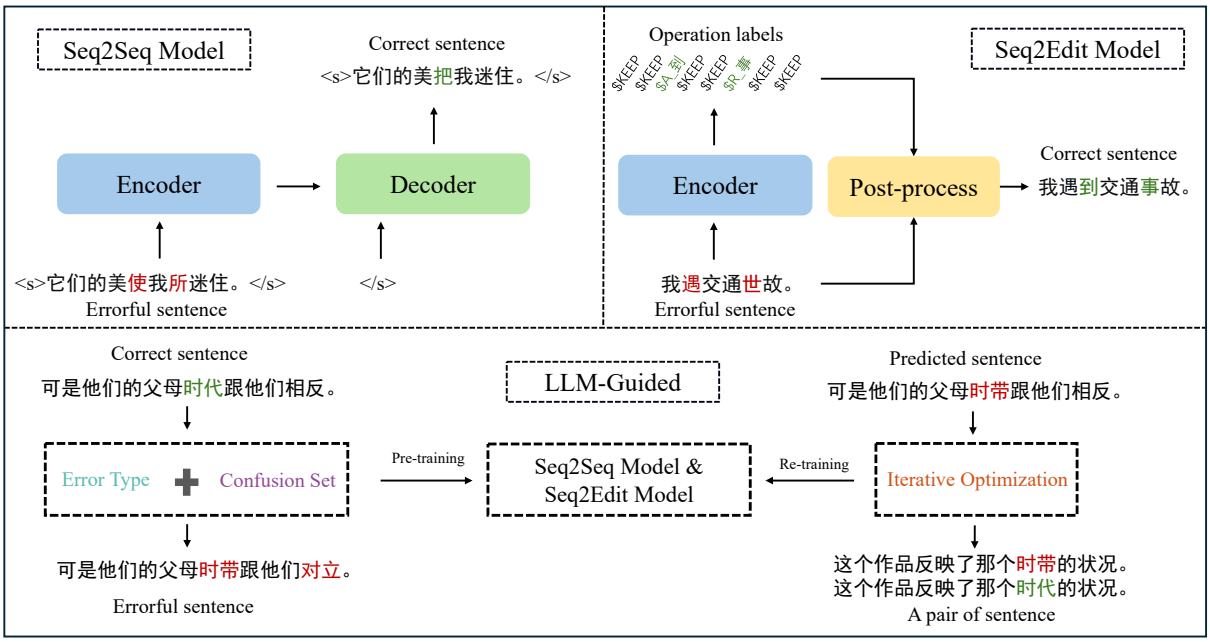


Figure 1: An example for CGEC corpus generation with LLMs.

al., 2023). More recently, large language models (LLMs) have demonstrated breakthrough performance in various natural language processing tasks (Wu et al., 2023; Fang et al., 2023). The LLMs benefit from a vast training corpus, providing them with ample grammatical knowledge, exceptional semantic comprehension, diversity generation, and strong representation capabilities (Brown et al., 2020; Zhou et al., 2023a). Several studies have investigated the possibility of LLMs for CGEC (Li et al., 2023b; Kaneko and Okazaki, 2023b). Research findings indicate that despite receiving supervised fine-tuning using CGEC data, the performance of LLMs on the CGEC task continues to be unsatisfactory (Zhang et al., 2023; Wu et al., 2023). The result may be due to the serious issue of overcorrection in LLMs. Recent research trends indicate that more comprehensive grammar information can be derived from LLMs (Song et al., 2023; Kaneko and Okazaki, 2023a). This grammar information can guide the GEC model to correct grammatical errors, enabling the model to correct complex errors that it was previously unable to correct. Due to the serious hallucination problem in LLMs, the corpus generated by LLMs may have slight deviations from real corpus. Therefore, simulating the distribution of grammatical errors in real-world scenarios and constructing high-quality training data remains a significant challenge.

To address this issue, we propose a large language model-guided optimization training method to enhance traditional GEC models with knowledge acquired from the LLMs. First, we utilize error types and the confusion set as extra knowledge to construct prompt instructions based on our given correct sentences, thus guiding LLMs to generate a diverse original synthetic corpus. Second, we directly fuse a few synthetic and reality corpora for the traditional GEC model training. Third, we utilize LLMs to analyze predicted sentences from GEC models, inferring whether there are any Chinese grammatical errors in the sentence that have not been corrected. Finally, we exploit LLMs to generate similar sentences based on the analysis results, which are used for re-training our GEC models iteratively, thus further enhancing the performance of GEC models.

Experiments on two benchmark datasets show that our proposed method achieves significant enhancements compared with the strong baselines, demonstrating that LLMs is extremely useful for optimizing the traditional GEC model training. Our thorough investigation highlights that error types and confusion sets are important factors in guiding LLMs to generate a diverse synthetic corpus, thus further expanding the error distribution of our training data. Detail comparison experiments show that re-training data generated by LLMs based on the inference and analysis results are essential for optimizing GEC model training since its error distributions are very similar to our testing data. The ablation study indicates

that both the early pseudo-corpus generation and the later iterative optimization are equally important in enhancing the performance of traditional GEC models, and they can benefit from each other.

2 Related Work

Grammatical error correction has been a challenging task in both natural language processing and machine learning fields. Recently, grammatical error correction has achieved significant improvement in English and Chinese. Now, we try to give a brief review of some representation approaches from two aspects, i.e., pre-trained language model-based methods and large language model-based methods.

Pre-trained language model-based methods. Motivated by the successful application of pre-trained language models on various artificial intelligence tasks, most previous works focus on enhancing the grammatical error correction performance by fusing the representations from pre-trained language models, which are mainly categorized into Seq2Seq and Seq2Edit two paradigms.

Seq2Seq models utilize encoder-decoder architectures to address the GEC task, which is drawn inspiration from neural machine translation (Kaneko et al., 2020; Zhang et al., 2022b; Zhou et al., 2023b). Later, Kaneko et al. (2020) extend the integration of pre-trained knowledge into the encoder-decoder models. Zhang et al. (2022b) utilizes syntactic information to enhance the model's performance. Zhou et al. (2023b) employ an external critic to appraise the suitability of the token to be produced gradually, impacting the selection of the succeeding token dynamically. For Seq2Edit models, they treat GEC as a sequence tagging task and predict token-level edit operations to be implemented on the input sentence (Omelianchuk et al., 2020; Zhang et al., 2022a). GECToR first iteratively annotates and revises the modified sequence, and then proceeds to correct it once more, thereby improving the speed of inference (Omelianchuk et al., 2020). Zhang et al. (2022a) integrate the GECToR model into the CGEC task and explore the impact of different pre-trained language models on this paradigm.

Compared with the Seq2Seq model, the Seq2Edit model has more efficient inference and robust error detection capabilities, but its performance has slightly declined (Li et al., 2022). In this work, we will give a detailed analysis of the influence of different training corpus on both Seq2Seq and Seq2Edit models.

Large language model-based methods. Recent researches on large language models indicate that increasing the size of language models can result in notable enhancements in performance across various downstream tasks (Kaplan et al., 2020). These LLMs have shown exceptional skill in adhering to and carrying out a variety of user instructions, leading to extensive acceptance among users who frequently engage with these models and utilize them for professional purposes (Brown et al., 2020; Dong et al., 2023; Li et al., 2023c; Zhao et al., 2023). Recently, there has been significant progress in corpus construction focused on reducing the manual annotation workload and automatically enhancing the scale, diversity, and creativity of corpus (Wang et al., 2023b; Asai et al., 2023; Zhou et al., 2023a). Wang et al. (2023b) provide an almost annotation-free method for aligning pre-trained language models with instructions. Asai et al. (2023) encourage LLMs to participate in self-reflective retrieval enhances the generation process, leading to improved quality and accuracy of language model output. Zhou et al. (2023a) design a small dataset containing 1000 samples that is used for training with standard supervised loss and demonstrate the potential of fine-tuning the LLMs on a high-quality small-scale dataset.

Recent work has explored the performance of various LLMs on the CGEC task (Fang et al., 2023; Qu and Wu, 2023; Fan et al., 2023; Zhang et al., 2023; Kaneko and Okazaki, 2023b; Song et al., 2023). Fang et al. (2023) evaluate the performance of ChatGPT on the CGEC task through in-context learning, highlighting its ability to produce coherent sentences while also exposing its tendency for excessive correction. Fan et al. (2023) employs ChatGPT to create ungrammatical sentences by incorporating specific clues. In cases where ungrammatical sentences lack clues, the author manually gathers such sentences from publicly accessible websites and rectifies them. Zhang et al. (2023) investigate the potential benefits of LLMs and enhance a writing-assistant scenario through multi-task instruction tuning. Despite supervised fine-tuning, the performance of LLMs remains unsatisfactory, indicating that they have not yet fully adapted to the GEC field. Consequently, researchers have started exploring if LLMs can play different roles in the GEC task, apart from merely acting as the corrector. Kaneko and Okazaki (2023b) propose a method to enhance the performance of the GEC task by leveraging LLMs for predicting edit

spans. Song et al. (2023) introduce a novel concept of elucidating grammar mistakes and demonstrate the capability of LLMs to provide explanations for grammatical errors.

In a word, although the notable advancements achieved by typical Seq2Seq and Seq2Edit methods in the task of CGEC, their effectiveness is limited by the quality and scale of the training corpus. Considering the strong capability of generation and inference demonstrated by LLMs, we will attempt to utilize LLMs to simulate the distribution of realistic errors and further optimize the typical Seq2Seq and Seq2Edit model training.

3 Our Proposed Approach

Large language model has shown its strong capability of generation and inference in various natural language processing tasks, such as question answer, summary generation, and knowledge inference (Brown et al., 2020; Zhou et al., 2023a; Zhang et al., 2023). However, since the parameters for LLMs are huge, it is difficult to fine-tune LLMs to adapt to each specific natural language processing task. Hence, the utilization of the ability of LLMs to enhance the performance of traditional GEC models is still a key challenge for Chinese grammatical error correction. To address this issue, we propose a large language model-guided optimization training method. On the one hand, we utilize the LLMs to generate a small-scale diverse pseudo corpus automatically based on error types and confusion sets, thus enlarging the error distributions of training data. On the other hand, we exploit the inference ability of LLMs to analyze the predicted results from the traditional GEC models, thus obtaining a more high-quality re-training corpus to make up for the limitations in traditional model training.

As depicted in Figure 2, our proposed method mainly contains three stages, i.e., *pseudo corpus generation*, *traditional model training*, and *iterative optimization*. First, we combine the error types, correct sentences, and an additional confusion set in one prompt to guide the LLMs in creating sentences with different grammatical errors. Second, the generated pseudo corpus is used to enlarge the original training data for traditional GEC model training. Third, we iteratively utilize LLMs to analyze predicted sentences by GEC models, inferring whether there are any Chinese grammar errors in the sentence that have not been corrected. Finally, we use LLMs to generate similar sentences based on the analysis results to retrain GEC models, further enhancing the model performances.

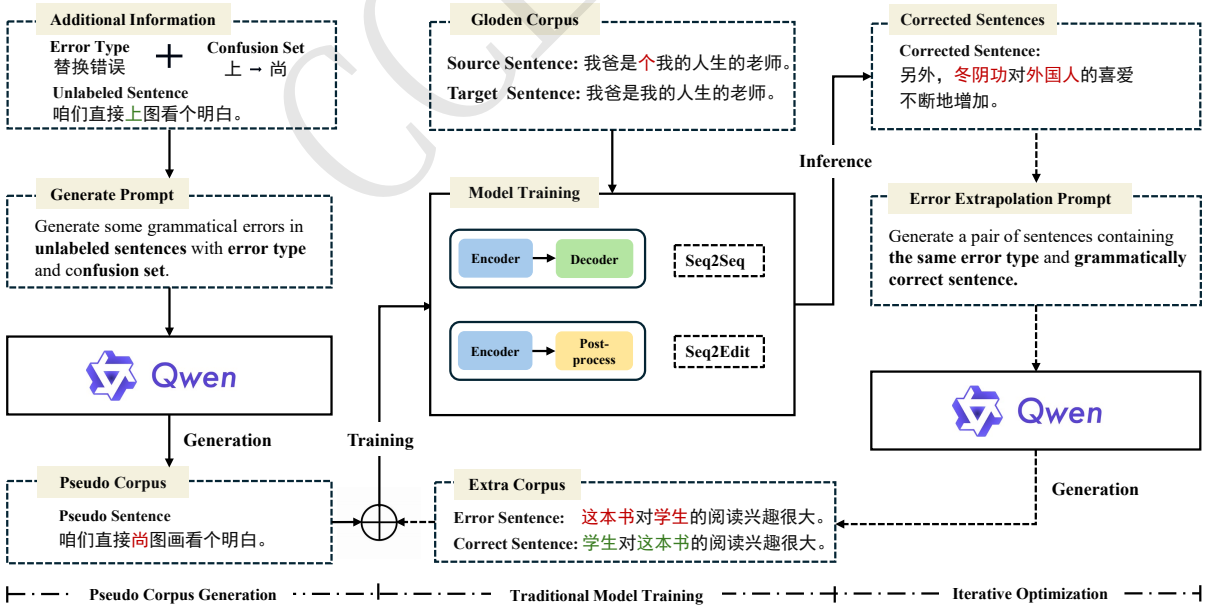


Figure 2: Framework of our proposed method.

3.1 Pseudo Corpus Generation

Traditional grammatical error correction models always rely on a large amount of high-quality parallel corpus to ensure effective performances. Therefore, how to automatically generate a high-quality corpus becomes an important factor in determining the model performance. Considering the strong generation capability of LLMs, we incorporate four error types and confusion set information as extra knowledge to constrain LLMs to generate a diverse pseudo corpus that covers all error types. In the practical experiments, we opt for the top 5 candidate characters that match each position within unlabeled sentences to maximize the utilization of the confusion set information. The process involves iterating through these candidates based on their probability in descending order. This selected candidate is then deemed as the final answer. The input of LLMs is formulated by converting error type E, confusion set information C, and unlabeled sentence Y into an instruction template $\mathcal{T}_{\text{gen}}(E, C, Y)$:

$$\mathcal{T}_{\text{gen}}(E, C, Y) = \{\overbrace{z_1, ..., z_{i-1}}^{instruction}, \overbrace{z_i, ..., z_{j-1}}^{E}, \overbrace{z_j, ..., z_k}^{C} \overbrace{z_{k+1}, ..., z_{k+n}}^{Y}\}. \tag{1}$$

Specifically, we first encourage LLMs to analyze grammatical details such as punctuation, vocabulary, and sentence structure in the correct sentences. Then, we utilize LLMs to accurately analyze the part-of-speech of each word in a given sentence. Next, specific error operations are performed in this sentence based on the analyzed part-of-speech and confusion set information. For instance, when encountering a substitution error, we will substitute partial words in the sentence according to the part-of-speech and confusion set information. Finally, LLMs are employed to identify Chinese grammar errors related to punctuation, vocabulary, and sentence structure, thereby verifying the presence of grammatical errors in the generated sentences. Through the above process, we restrict the generation of pseudo-corpus by LLMs using error types and confusion set information. This "constrain-then-generate" approach enables us to generate more diverse, informative, and realistic pseudo instances.

3.2 Traditional Model Training

Given a sentence $X = \{x_1, x_2, \ldots, x_m\}$ that may include grammatical errors, the CGEC task aims to identify and correct the potential grammatical errors in X and outputs the corresponding error-free sentence $Y = \{y_1, y_2, \ldots, y_n\}$.

Seq2Seq Model. The Seq2Seq model employs the encoder-decoder framework to generate the correct sentence based on the ungrammatical input sentence. It uses a multi-layer self-attention as the encoder module to encode the entire erroneous sentence $x_1, x_2, \ldots, x_m$ into the corresponding hidden states $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_m$. Then, another multi-layer cross-attention is used as the decoder module to generate each token in $y_1, y_2, \ldots, y_n$ autoregressively based on the hidden states and previously generated tokens. The training objective of the Seq2Seq model is to minimize the negative log-likelihood loss:

$$\mathcal{L}(\theta) = -\sum_{i=1}^{n} \log p(\hat{y}_i = y_i | X, Y_{<i}, \theta), \tag{2}$$

where θ is learnable model parameters, $\hat{y}_i$ is the i-th token predicted by the model, and $Y_{<i} = y_1, y_2, ..., y_{i-1}$ denotes a set of tokens before the i-th token y_i.

Seq2Edit Model. Due to the overlap between X and Y, it is inefficient to perform autoregressive generation on the entire target Y, while the Seq2Edit model is a good choice. The Seq2Edit model typically regards GEC as a sequence labeling task, which is composed of a BERT-like encoder and a simple classifier stacked on the top. Firstly, a pre-defined set of tags is needed to represent editing operations. In general, this set of tags includes generic edits (e.g.*$KEEP* for keeping the current token unchanged, *$DELETE* for deleting the current token) and token-dependent edits(e.g. *$APPEND* for appending a new token next to the current token, *$REPLACE* for replacing the current token with another token). Considering that the vocabulary of tags linearly increases with token-dependent edits, it is common to set a moderate tag vocabulary based on the frequency of editing to balance the coverage of editing and model size. Then, the initial sentence pairs are converted into sentence-tag pairs of identical length. Specifically, align the target sentence Y with the source sentence X by minimizing the modified Levenshtein

distance, and then convert it into a sequence of tags $T = t_1, t_2, ..., t_m$. In training, the general objective function of the Seq2Edit methods is to minimize the negative log-likelihood loss for the tag sequence:

$$\mathcal{L}(\theta) = -\sum_{i=1}^{n} \log p(\hat{t}_i = t_i | X, \theta), \tag{3}$$

where $\hat{t}_i$ is the i-th tag predicted by the model. During the inference stage, Seq2Edit methods predict a tagging sequence T, then perform editing operations on the source sentence X via post-processing to obtain the predicted output Y.

3.3 Iterative Optimization

Since traditional GEC models are sensitive to some high-error types with high-probability distributions but ignore low-probability ones, we first employ LLMs to analyze grammar errors that these models did not rectify. Then, we exploit LLMs to generate sentences containing uncorrected errors based on the analysis results. Finally, the newly generated corpus is used to iteratively re-train traditional GEC models, thus further enhancing their accuracy.

In the iterative optimization process, we incorporate two different data sources: the pseudo corpus $\mathcal{D}_{\mathtt{pse}}$, and a small human-annotated corpus known as golden data, represented as $\mathcal{D}_{\mathtt{ann}}$. To bridge the distribution gap between the pseudo corpus and the human-annotated corpus, we make use of $\mathcal{D}_{\mathtt{ann}}^{\mathtt{dev}}$. Subsequently, $\mathcal{D}_{\mathtt{ann}}^{\mathtt{test}}$ is employed to evaluate the efficiency of GEC models.

Algorithm 1 An algorithm with Iterative Optimization

Input: Pseudo corpus $\mathcal{D}_{\mathtt{pse}}$; Development dataset $\mathcal{D}_{\mathtt{ann}}^{\mathtt{dev}}$; GEC models $\mathtt{f}$;
The LLM model $\mathbb{P}_{\mathtt{LLM}}$; The number of iterative optimization rounds $\mathtt{N}$;
Iterative optimization prompts $\mathcal{T}_{\mathtt{IE}}$.
Output: Re-training corpus $\mathcal{D}_{\mathtt{retrain}}$; Optimized GEC models $\mathtt{f}'$.

$\mathcal{D}_{\mathtt{add}}^{0} \leftarrow \emptyset$
for `r in range(N)` **do**
  `Init(f);` ▷ Initialize GEC models with annotated training corpus
  $\mathcal{D}_{\mathtt{retrain}}^{\mathtt{r}} \leftarrow \mathcal{D}_{\mathtt{pse}} \cup (\cup_{\mathtt{i=1}}^{\mathtt{r}} \mathcal{D}_{\mathtt{add}}^{\mathtt{i}})$
  `Retraining(f,`$\mathcal{D}_{\mathtt{retrain}}^{\mathtt{r}}$`)`
  $\mathcal{D}_{\mathtt{pre}}^{\mathtt{r}} \leftarrow \mathtt{Correct}\{\mathtt{f}'(\mathcal{D}_{\mathtt{ann}}^{\mathtt{dev}} \mid \mathcal{D}_{\mathtt{retrain}}^{\mathtt{r}})\}$ ▷ Evaluate GEC models with synthetic corpus
  $\mathcal{D}_{\mathtt{add}}^{\mathtt{r+1}} \leftarrow \emptyset$
  for `each (x,y)` $\in \mathcal{D}_{\mathtt{pre}}^{\mathtt{r}}$ **do**
    $(\mathtt{x}_{\mathtt{add}}, \mathtt{y}_{\mathtt{add}}) \sim \mathbb{P}_{\mathtt{LLM}}(\cdot | \mathcal{T}_{\mathtt{IE}}(\mathtt{x}_{\mathtt{pre}}, \mathtt{y}_{\mathtt{ann}}))$ ▷ Identify errors and generate coupus
    $\mathcal{D}_{\mathtt{add}}^{\mathtt{r+1}} \leftarrow \mathcal{D}_{\mathtt{add}}^{\mathtt{r+1}} \cup \{(\mathtt{x}_{\mathtt{add}}, \mathtt{y}_{\mathtt{add}})\}$
  end for
end for
$\mathcal{D}_{\mathtt{retrain}} \leftarrow \mathcal{D}_{\mathtt{pse}} \cup (\cup_{\mathtt{i=1}}^{\mathtt{N}} \mathcal{D}_{\mathtt{add}}^{\mathtt{i}})$ ▷ Iteration completion and hybridize corpus
$\mathtt{f}' = \mathtt{Re\text{-}training(f,}\mathcal{D}_{\mathtt{retrain}}\mathtt{)}$

We present the whole process of iterative optimization in Algorithm 1. First, we set the additional corpus $\mathcal{D}_{\mathtt{add}}^{0}$ to an empty set in the 0-th round. The iterative optimization process starts in the second line, represented by a for-loop, where each iteration of the for-loop corresponds to one optimization round. In our experiment, we conduct 4 rounds of iterative optimization tests. For each loop, the parameters of GEC models are initialized with an annotated training corpus. Second, we combine the pseudo corpus $\mathcal{D}_{\mathtt{pse}}$ and the additional corpus $\mathcal{D}_{\mathtt{add}}^{\mathtt{r}}$ from the previous r-th rounds to form the re-training corpus $\mathcal{D}_{\mathtt{retrain}}^{\mathtt{r}}$ in the r-th round. Then, all GEC model parameters are updated by the re-training corpus $\mathcal{D}_{\mathtt{retrain}}^{\mathtt{r}}$. Third, we evaluate optimized GEC models $\mathtt{f}'$ on the development dataset $\mathcal{D}_{\mathtt{ann}}^{\mathtt{dev}}$ and combine the predicted sentences $\mathtt{x}_{\mathtt{pre}}$ of GEC models to form the set of predicted sentences $\mathcal{D}_{\mathtt{pre}}^{\mathtt{r}}$ in the r-th round. We set the additional corpus $\mathcal{D}_{\mathtt{add}}^{\mathtt{r+1}}$ in the `r+1`-th round to an empty set. Fourth, we utilize $\mathcal{T}_{\mathtt{IE}}(\mathtt{x}_{\mathtt{pre}}, \mathtt{y}_{\mathtt{ann}})$ prompt to motivate the LLMs to analyze whether there are still grammatical errors in

the predicted sentences x_{pre} through optimized GEC models. Based on the analysis results, we employ the LLMs generate sentence paris (x_{add}, y_{add}) resembling the predicted sentence x_{pre} with the annotated sentence y_{ann}. Finally, We combine the pair of generated sentence (x_{add}, y_{add}) to form the additional corpus $\mathcal{D}_{add}^{r+1}$ of the $r+1$ round. After N iterative optimizations, we hybridize the generated additional corpus $\mathcal{D}_{add}^{N}$ with the pseudo corpus $\mathcal{D}_{pse}$ to form the ultimate re-training corpus $\mathcal{D}_{retrain}$. Then, we retrain GEC models based on the ultimate re-training corpus $\mathcal{D}_{retrain}$ to obtain the final iteratively optimized GEC models. The primary steps of the iterative optimization algorithm involve retraining the GEC models using the current synthesized corpus (line 5) and employing the LLM to extend the predicted sentences to generate additional training corpus (lines 8-11). In summary, the iterative optimization process utilizes LLMs to identify errors within the GEC models and generate a corpus based on this error information, thus helping GEC models find better convergence points.

4 Experiments

4.1 Experimental Settings

Datasets. We adopt the wudao corpus[0] as the unlabeled corpus to generate pseudo corpus for the large language model-guided optimization training method. We employ a combination of the Chinese Lang8 (Zhao et al., 2018a), the HSK (Zhang, 2009), and our generated pseudo corpus by LLMs for traditional model training. MuCGEC-*dev* (Zhang et al., 2022a) is used for hyper-parameter tuning and checkpoint selection. Table1 presents detailed statistical information for the mentioned datasets.

Dataset	#Sent	#Error	Usage
Lang8	1,220,906	1,092,285 (89.5%)	Training
HSK	156,870	95,320 (60.8%)	Training
WuDao	200,000	-	Training
MuCGEC-*dev*	1,125	1,069 (95.1%)	Validation
MuCGEC-*test*	5,938	5,475 (92.2%)	Testing
NLPCC2018-*test*	2,000	1,983 (99.2%)	Testing

Table 1: Statistics of the used CGEC datasets. **#Sent** denotes the number of the sentences and **#Error** denotes the number (the percentage) of the erroneous sentences.

Evaluation Metrics. For the evaluation metrics, we follow previous work and report word-level performance on the NLPCC2018-*test* using the official MaxMatch scorer and the PKUNLP word segmentation tool provided by Zhao et al. (2018b). For MuCGCE-*test* (Zhang et al., 2022a), we report the character-level scores using the ChERRANT tools.

Basic Model. The mainstream CGEC models are primarily categorized into two types, i.e., Seq2Seq and Seq2Edit. We have chosen the Seq2Seq and Seq2Edit models as the baseline representatives. For the Seq2Seq model, we opt for Chinese BART-large (Shao et al., 2021) to generate text, as it can be straightforwardly trained for CGEC. For the Seq2Edit model, GECToR-Chinese (Omelianchuk et al., 2020) stands out as the predominant method employed for CGEC. Building upon the foundation laid by Zhang et al. (2022a), our selection for the encoder in the Seq2Edit method is Chinese-Struct-Bert-Large (Wang et al., 2020). We employ Qwen-14B-Chat (Bai et al., 2023)[1] for pseudo corpus generation and Qwen-72B-Chat-Int4 (Bai et al., 2023)[2] for iterative optimization.

Implementation Details. We utilize Chinese BART-large (Shao et al., 2021) and Chinese-Struct-Bert-Large (Wang et al., 2020) to enhance traditional GEC models. For the LLMs inference process, we follow the vllm-project (Kwon et al., 2023)[3] on 4 GeForce RTX 3090 24GB GPUs. Appendix A

[0]https://data.baai.ac.cn/details/WuDaoCorporaText
[1]https://huggingface.co/Qwen/Qwen-14B-Chat
[2]https://huggingface.co/Qwen/Qwen-72B-Chat-Int4
[3]https://github.com/vllm-project/vllm

shows additional implementation specifics and hyperparameter selection. Appendix B gives detailed illustrations about particular prompts utilized in our approach.

4.2 Main Results

System	Extra Data Size	NLPCC2018-*test* P	NLPCC2018-*test* R	NLPCC2018-*test* $F_{0.5}$	MuCGEC-*test* P	MuCGEC-*test* R	MuCGEC-*test* $F_{0.5}$
Seq2Edit (Zhang et al., 2022a)	-	42.88	30.19	39.55	44.11	27.18	39.22
Seq2Seq (Zhang et al., 2022a)	-	41.44	32.89	39.39	43.81	28.56	39.58
SynGEC (Zhang et al., 2022b)	-	49.96	33.04	45.32	54.69	29.10	46.51
TemplateGEC (Li et al., 2023a)	-	54.50	27.40	45.50	-	-	-
Qwen-14B-Chat (Bai et al., 2023)	-	24.50	41.71	26.70	22.47	34.85	24.19
Qwen-72B-Chat-Int4 (Bai et al., 2023)	-	27.94	33.11	28.84	25.62	30.41	26.45
Direct Noise (Ye et al., 2023)	800w	49.57	31.80	44.59	54.93	29.61	46.91
Backtranslation (Ye et al., 2023)	800w	47.64	36.43	44.88	54.82	30.27	47.17
Our Seq2Edit	-	42.87	30.25	39.57	43.97	27.45	39.25
***w/* Iterative Optimization**	20w	43.74	36.61	**42.10**	44.34	32.60	**41.36**
Our Seq2Seq	-	49.68	31.79	44.65	54.17	28.74	46.03
***w/* Iterative Optimization**	20w	52.17	31.51	**46.12**	55.74	29.56	**47.35**

Table 2: Overall results on NLPCC2018-*test* and MuCGEC-*test*. The results of Seq2Edit (Zhang et al., 2022a), Seq2Seq (Zhang et al., 2022a), SynGEC (Zhang et al., 2022b), TemplateGEC (Li et al., 2023a), Direct Noise and Backtranslation (Ye et al., 2023) on NLPCC2018-*test* and MuCGEC-*test* are cited from the original papers and other results including Qwen-14B-Chat and Qwen-72B-Chat (Bai et al., 2023) are implemented by us. The best results of two different traditional models are highlighted in bold.

Table 2 shows the main results of NLPCC2018-*test* and MuCGEC-*test* and makes a detailed comparison with previous works. First, we can see that our Seq2Seq and Seq2Edit models have achieved comparable or even better performances than previous works (Zhang et al., 2022a), which verifies the strong ability of our baseline models. Second, compared with the top block, we find that previous methods frequently integrate supplementary information to boost the efficacy of GEC models, including syntactic and confusion set knowledge (Zhang et al., 2022a; Li et al., 2023a). Due to LLMs acquiring a strong understanding of grammar during training, our approach focuses on effectively excavating this knowledge from LLMs and utilizing it to enhance the performance of GEC models. Our approach consistently surpasses all comparison methods in terms of $F_{0.5}$. This result indicates that our method can effectively excavate the grammar knowledge from LLMs and utilize it in Seq2Seq and Seq2Edit models. Third, we find that our large language model-guided iterative optimization training approach leads to 1.47/1.32 improvements in the Seq2Seq model and 2.53/2.11 improvements in the Seq2Edit model compared to the traditional training method. Compared with direct noise and backtranslation augmentation methods, our model can achieve better results with only a smaller pseudo corpus. This highlights the efficacy of our iterative optimization strategy and the small-scale corpus construction method we introduced. Finally, we can observe that our approach performs superior results with Seq2Seq and Seq2Edit models compared to utilizing LLMs directly for grammatical error correction, indicating that employing constraints like error types and confusion sets can guide LLMs to construct corpus effectively. Moreover, we can effectively utilize the strong analytical capabilities of LLMs to enhance the performance of Seq2Seq and Seq2Edit models.

Proceedings of the 23rd China National Conference on Computational Linguistics, pages 1366-1380, Taiyuan, China, July 25 - 28, 2024.
Volume 1: Main Conference Papers

4.3 Analyses and Discussion

4.3.1 Ablation Study

The main results of our approach stem from three kinds of constraint information, including error types, confusion set, and iterative optimization from LLMs. Hence, it is imperative for us to perform ablation studies on these three types of grammar information to assess their impacts on our method. Table 3 shows a thorough ablation study on the NLPCC2018-*test* and MuCGEC-*test* datasets.

Method	NLPCC2018-*test*			MuCGEC-*test*		
	P	**R**	$F_{0.5}$	**P**	**R**	$F_{0.5}$
Seq2Edit	43.74	**36.61**	**42.10**	44.34	**32.60**	**41.36**
w/o Error Types	43.52	35.32	41.59	43.97	32.06	40.93
w/o Confusion Set	**43.92**	35.11	41.82	**44.47**	31.39	41.05
w/o Iterative Optimization	42.87	36.10	41.32	43.74	32.12	40.79
Seq2Seq	51.82	32.03	**46.12**	55.74	29.56	**47.35**
w/o Error Types	51.41	31.40	45.60	55.23	29.33	46.94
w/o Confusion Set	**51.96**	31.30	45.90	**55.83**	28.86	47.04
w/o Iterative Optimization	49.87	**33.75**	45.52	54.42	**30.06**	46.83

Table 3: Results of ablation study on NLPCC2018-*test* and MuCGEC-*test*

First, we can observe that incorporating each type of constraint information leads to notable enhancements in both Seq2Seq and Seq2Edit models when implemented separately, affirming the validity of our decision to acquire data from LLMs. Due to the high proportion of substituted errors in the HSK corpus, the enhancement in the performance of the GEC model does not vary significantly based on error types and constraints related to confusion set information. The primary cause of this result is that the model becomes trapped in spurious patterns during the training process, leading to minimal performance enhancements. Therefore, iterative optimization has brought the greatest improvement for Seq2Seq and Seq2Edit models, indicating that LLMs based on corrected sentences and grammatically correct sentences can generate valuable reference corpora. These outcomes offer valuable insights for Seq2Seq and Seq2Edit models, resulting in substantial benefits for these models. Moreover, the combination of different constraint information leads to sustaining improvements in all GEC models, demonstrating the compatibility between the three types of our designed information.

4.3.2 Distribution on four error types

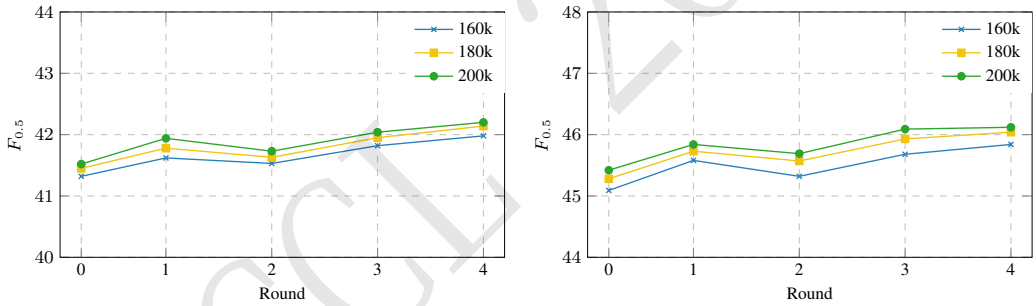
To further verify the effectiveness of our method in narrowing the gap in error type distribution compared to the golden corpus, we utilize the HSK corpus as the foundation corpus. The error type distribution is illustrated across the four CGEC error categories: missing (M), redundant (R), substitution (S), and word-order (W). We utilize the ChERRANT tool for converting the corpus into the M2 file format and extracting the distribution of error types within the corpus.

The results depicted in Figure 3 demonstrate that our method further narrows the error type distribution gap between the pseudo corpus and the realistic corpus. The constraint information in the confusion set primarily focuses on addressing substituted errors. Therefore, we only replace the part of the corpus that consists of substitution errors generated by error type constraint information. Through iterative optimization information, in cases where the proportion of missing errors is relatively low, we iteratively generate more corpus containing missing errors to further narrow the error type distribution gap between the generated corpora and the realistic corpora. In brief, our method closely aligns with the error type distribution found in the HSK corpus by utilizing constraints that take into account error types, confusion sets, and iterative optimization.

Figure 3: The results of distribution on four error types

4.3.3 Effectiveness of Iterative Optimization

To further investigate the effect of our iterative optimization method on traditional GEC models, we construct three different scales of pseudo corpus (160k, 180k, and 200k sentences) based on error types and confusion sets constraint information, and integrate them into the Lang8 and HSK corpora. Then, we train the GEC model on these three different scales of corpus and perform four rounds of iterative optimization on the NLPCC2018 test set. In other words, we want to use the powerful analytical capabilities of LLMs to analyze whether there are any remaining uncorrected grammatical errors in sentences predicted by the GEC model. Based on the analysis results, we aim to generate a high-quality corpus that is instructive.



(a) Seq2Edit (b) Seq2Seq

Figure 4: Results of our method on NLPCC2018 test set with different pseudo corpus scales.

By observing the results in Figure 4, we find that when grammatically correct sentences are added to LLMs to generate re-training data, it helps to improve the performance of the GEC model. As the number of iterations and corpus size grows, the performance of the GEC model continues to improve. This also indicates that enabling LLMs to view golden sentences during the generation of pseudo corpus enables them to capture the grammatical information from incorrect sentences to correct ones, thereby reducing the probability of GEC model falling into a pattern of illusory modifications. Therefore, we can also understand why there is a significant performance improvement when golden sentences are added to LLM inputs during the generation of training data. In this scenario, LLMs generate sentences in the training data similar to golden sentences in the test set.

5 Conclusion

In this paper, we propose a large language model-guided optimization training method to exploit the strong generation and inference capability of LLMs to enhance the performance of traditional GEC

models. On the one hand, we propose using error types and confusion sets as constraint information to guide LLMs in generating diverse pseudo corpus, thus enhancing the robustness of traditional GEC models. On the other hand, we leverage LLMs to infer the predictions of our GEC models to reduce the gap between the error distributions of re-training and realistic corpora. iteratively optimizing our CGEC models. Experimental results on two CGEC datasets demonstrate the effectiveness of our approach on Seq2Edit and Seq2Seq models. Further detailed analysis shows the effectiveness of our proposed method in reducing the distribution of error types, as well as the key role of iterative optimization in enhancing performance. In future research, we will further explore how to fully utilize the grammatical information of large models to enhance the performance of Seq2Seq and Seq2Edit models.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (U21B2027, 62366027, 62266028, 62306129), Yunnan Provincial Major Science and Technology Special Plan Projects (202103AA080015, 202202AD080003, 202203AA080004), Yunnan Fundamental Research Projects (202401BC070021, 202301AS070047, 202401CF070121), Kunming University of Science and Technology's "Double First-rate" Construction Joint Project (202201BE070001-021, 202301BE070001-027), Yunnan High and New Technology Industry Project (201606). The authors would like to thank the anonymous reviewers for their valuable comments and suggestions for improving this paper.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *CoRR*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *CoRR*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proc. of NeurIPS*.

Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe. 2022. Grammatical error correction: A survey of the state of the art. *CoRR*.

Yo Joong Choe, Jiyeon Ham, Kyubyong Park, and Yeoil Yoon. 2019. A neural grammatical error correction system built on better pre-training and sequential transfer learning. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 213–227.

Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang. 2023. A survey of natural language generation. *ACM Comput. Surv.*, pages 173:1–173:38.

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 *Proc. of NLPCC*, pages 69–80.

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*.

Masahiro Kaneko and Naoaki Okazaki. 2023a. Controlled generation with prompt insertion for natural language explanations in grammatical error correction. *CoRR*.

Masahiro Kaneko and Naoaki Okazaki. 2023b. Reducing sequence length by predicting edit spans with large language models. In *Proc. of EMNLP*, pages 10017–10029.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proc. of ACL*, pages 4248–4254.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*.

Shota Koyama, Hiroya Takamura, and Naoaki Okazaki. 2021. Various errors improve neural grammatical error correction. In *Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation, PACLIC 2021, Shanghai International Studies University, Shanghai, China, 5-7 November 2021*, pages 251–261.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Jiquan Li, Junliang Guo, Yongxin Zhu, Xin Sheng, Deqiang Jiang, Bo Ren, and Linli Xu. 2022. Sequence-to-action: Grammatical error correction with action guided sequence generation. In *Proc. of AAAI*, pages 10974–10982.

Yinghao Li, Xuebo Liu, Shuo Wang, Peiyuan Gong, Derek F. Wong, Yang Gao, Heyan 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. *CoRR*.

Yinghui Li, Zishan Xu, Shaoshen Chen, Haojing Huang, Yangning Li, Yong Jiang, Zhongli Li, Qingyu Zhou, Hai-Tao Zheng, and Ying Shen. 2023c. Towards real-world writing assistance: A chinese character checking benchmark with faked and misspelled characters. *CoRR*.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem N. Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector - grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2020, Online, July 10, 2020*, pages 163–170.

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

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. CPT: A pre-trained unbalanced transformer for both chinese language understanding and generation. *CoRR*.

Yixiao Song, Kalpesh Krishna, Rajesh Bhatt, Kevin Gimpel, and Mohit Iyyer. 2023. Gee! grammar error explanation with large language models. *CoRR*.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proc. of EACL*, pages 37–47.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Jiangnan Xia, Zuyi Bao, Liwei Peng, and Luo Si. 2020. Structbert: Incorporating language structures into pre-training for deep language understanding. In *Proc. of ICLR*.

Yu Wang, Yuelin Wang, Kai Dang, Jie Liu, and Zhuo Liu. 2021. A comprehensive survey of grammatical error correction. *ACM Trans. Intell. Syst. Technol.*, pages 65:1–65:51.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, et al. 2023a. Instructuie: Multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Proc. of ACL*, pages 13484–13508.

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

Jingheng Ye, Yinghui Li, Yangning Li, and Hai-Tao Zheng. 2023. Mixedit: Revisiting data augmentation and beyond for grammatical error correction. In *Proc. of EMNLP Findings*, pages 10161–10175.

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 *Proc. of NAACL*, 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 *Proc. of EMNLP*, pages 2518–2531.

Yue Zhang, Leyang Cui, Deng Cai, Xinting Huang, Tao Fang, and Wei Bi. 2023. Multi-task instruction tuning of llama for specific scenarios: A preliminary study on writing assistance. *CoRR*.

Baolin Zhang. 2009. Features and functions of the hsk dynamic composition corpus. *International Chinese Language Education*, pages 71–79.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018a. 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*, pages 439–445.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018b. Overview of the NLPCC 2018 shared task: Grammatical error correction. In *Proc. of NLPCC*, pages 439–445.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. LIMA: less is more for alignment. In *Proc. of NeurIPS*.

Houquan Zhou, Yumeng Liu, Zhenghua Li, Min Zhang, Bo Zhang, Chen Li, Ji Zhang, and Fei Huang. 2023b. Improving seq2seq grammatical error correction via decoding interventions. In *Proc. of EMNLP Findings*, pages 7393–7405.

A Implementation Details and Hyperparameters

For Seq2Edit model and Seq2Seq model training, following Zhang et al. (2022a), we list the main hyper-parameters in Table 4 and Table 5. To determine the optimal balancing weights for the training objective, we experiment with various values for α within 0.5, 1.0, 1.2, 2.0 and β within 0.5, 1.0, 2.0. Our experiments reveal that the configuration with α = 1.0 and β = 1.0 achieves the highest $F_0.5$ score on NLPCC2018-*test* and MuCGEC-*test*, which is used as the default settings in all our experiments.

Configurations	Values
Backbone	Chinese-StructBERT
Devices	1 GeForce RTX 3090
Number of max epochs	20
Number of cold epochs	2
Optimizer	Adam
Cold learning rate	1×10^{-3}
Learning rate	1×10^{-5}
Batch size	128
Patience	3

Table 4: Hyper-parameter values of Seq2Edit.

Configurations	Values
Backbone	Chinese-BART-large
Devices	4 GeForce RTX 3090
Epochs	20
Optimizer	Adam
Learning rate	3×10^{-5}
Batch size per GPU	4096 tokens
Warmup updates	2000
Max source length	1024
Dropout	0.2
Dropout-src	0.2
Beam size	12

Table 5: Hyper-parameter values of Seq2Seq.

B Designed Prompts

In order to guide LLMs to generate high-quality corpus under the constraints of error types and confusion set, we have carefully designed instruction prompts based on the characteristics of constructing the CGEC task corpus. The prompt for generation is shown in Table 6 and the prompt for iterative optimization is shown in Table 7.

We mainly cover missing, redundant, substitution, and word-order as the primary types of errors. For missing errors, we prompt the LLMs to delete one or more words in the input sentence. For redundant errors, we prompt the LLMs to repeat one or more words in the input sentence, but only from the words present in the input sentence. For substitution errors, we prompt the LLMs to replace one or more of the words in the input sentence with synonyms, homophones, or words that are similar in form to that word. For word-order errors, we prompt the LLMs to rearrange the order of words in the input sentence, resulting in a disorganized sentence structure. It is important to keep the words in the sentence unchanged. The corpus generation methods for the various error types mentioned above correspond to the "operation" section of the prompt for constructing corpus.

你的目标是针对输入句子的标点、词语和句法等，在其中巧妙地引入实用、合理且忠实的中文语法错误。实用性要求构造的句子有助于提升中文语法纠错系统的性能和帮助人们理解正确的中文语法规则。合理性要求构造后的句子能被中文语法纠错系统纠正。忠实性要求构造方式是基于指定的中文错误类型和混淆集信息。 指定的中文语法错误类型为[error_types]。请遵循[工作方式]，严格保持句子编号不变。下面让我们一步步地按照[步骤]，针对[输入句子]，生成中文语法错误的句子。 [混淆集信息] {con_info} [步骤] Step1: 初步分析输入句子的标点、词语和句法等。 Step2: 准确分析出输入句子中每个词语的词性。 Step3: 根据分析的词性、中文语法错误类型和混淆集信息，针对输入句子中的词语，{operations}。 Step4: 分析一下构造句子中的标点、词语和句法等，确保句子中包含中文语法错误。 Step5: 无需进行额外的解释，直接输出构造的句子。 [工作方式] [ID][输入句子]<中文语法正确的句子。> [ID][输出句子]<构造的句子。> 请严格遵循你的工作方式。现在开始对输入的句子进行改动： [ID][输入句子]<input_sentence>

Table 6: Our designed prompt for constructing corpus.

你的目标是分析纠错模型预测的句子中的标点、词语和句法等，判断其是否含有未纠正的中文语法错误。如果有，生成一对实用、合理且忠实的句子对。如果没有，则回复[纠错完全]。实用性要求构造的句子有助于提升中文语法纠错系统的性能和帮助人们理解正确的中文语法规则。合理性要求构造后的句子能被中文语法纠错系统纠正。忠实性要求基于未纠正的中文语法错误进行构造。
请遵循[工作方式]，严格保持句子编号不变。下面让我们一步步地按照[步骤]根据[预测句子]和[语法正确句子]的内容生成句子对，无需输出解释过程，直接给出结果。
[步骤]
Step1: 初步分析纠错模型预测后句子中的标点、词语和句法等。
Step2: 准确分析出纠错模型预测后句子中每个词语的词性。
Step3: 根据分析的词性和语法正确句子，判断预测句子中是否还含有未纠正的中文语法错误。如果有则执行下面的步骤，否则回复[纠错完全]。
Step4: 生成类似句子对，其包含语法纠错模型未纠正的中文语法错误的句子和正确句子。
Step5: 检查生成的句子对，保证其包含纠错模型未纠正的中文语法错误。
[工作方式]
[ID][预测句子]<被中文语法纠错模型预测的句子。>
[ID][语法正确句子]<对应语法完全正确的句子。>
[ID][构造的错误句子]<包含语法纠错模型未纠正的中文语法错误的句子。>
[ID][构造的正确句子]<中文语法正确的句子。>
请严格遵循你的工作方式。现在开始根据预测句子和语法正确句子生成类似的句子对：
[ID][预测句子]<predicted_sentence>
[ID][语法正确句子]<annocated_sentence>

Table 7: Our designed prompt for iterative optimization.