Detection-Correction Structure via General Language Model for Grammatical Error Correction

Anonymous ACL submission

Abstract

Grammatical error correction (GEC) is a task dedicated to rectifying texts with minimal edits, which can be decoupled into two components: detection and correction. However, previous works have predominantly focused on direct correction, with no prior efforts to integrate both into a single model. Moreover, the exploration of the detection-correction paradigm by large language models (LLMs) remains underdeveloped. This paper introduces an integrated detection-correction structure, named DeCoGLM, based on the General Language Model (GLM). The detection phase employs a fault-tolerant detection template, while the correction phase leverages autoregressive mask infilling for localized error correction. Through the strategic organization of input tokens and modification of attention masks, we facilitate multi-task learning within a single model. Our model demonstrates competitive performance against the state-of-the-art models on English and Chinese GEC datasets. Further experiments present the effectiveness of the detection-correction structure in LLMs, suggesting a promising direction for GEC.

1 Introduction

Grammatical error correction (GEC) is a task focused on automatically rectifying grammatical errors in human-written text (Wang et al., 2021). GEC models are applied in language learning (Katinskaia and Yangarber, 2021; Caines et al., 2023; Kaneko et al., 2022), enhancing automatic speech recognition (Liao et al., 2023), and aiding in text data labeling (Sun et al., 2023). The two primary approaches in GEC are Sequence-to-Sequence (Seq2Seq) and Sequence-to-Edit (Seq2Edit). Without detection, Seq2Seq treats GEC as the direct generation for correct text, providing high flexibility (Junczys-Dowmunt et al., 2018; Ge et al., 2018). On the other hand, Seq2Edit views GEC as a sequence labeling task for edit labels, showcasing high precision by controlled edits (Awasthi et al., 2019; Stahlberg and Kumar, 2020; Omelianchuk et al., 2020). The advent of large language models (LLMs) has further expanded Seq2Seq model capabilities (Ouyang et al., 2022; Zeng et al., 2022). Despite their unprecedented performance in various tasks (Chang et al., 2024), LLMs underperform than low-parameter models in GEC due to the over-correction phenomenon (Qu and Wu, 2023; Coyne et al., 2023).

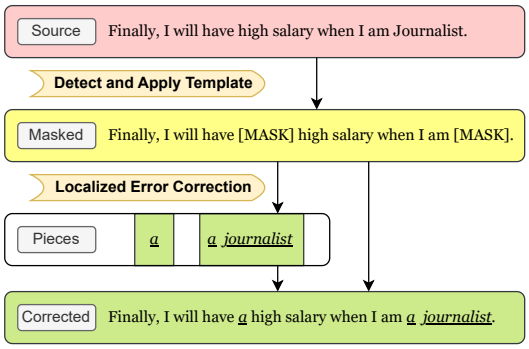

Figure 1: Detection and correction process of DeCoGLM. **Detection** and **Correction** are incorporated in one General Language Model (GLM).

While the detection-correction structure can harness the strengths of both Seq2Seq and Seq2Edit, most existing works merely utilize detection as additional input for Seq2Seq models (Yuan et al., 2021a; Li et al., 2022, 2023a). Moreover, all previous detection-correction systems comprise separate models (Chen et al., 2020). In contrast, we introduce a novel GEC model, named DeCoGLM, based on the General Language Model (GLM) (Du et al., 2022). This model employs an integrated detection-correction structure to detect errors and generate localized corrections. As depicted in Figure 1, the error detection phase employs a template rule to construct masked text based on detection results. During the correction phase, the model leverages the autoregressive mask infilling capabil-

ity of the GLM to generate correct text pieces for erroneous parts, thereby saving inference time. To incorporate both detection and correction within a single model, we devise a multi-task learning approach, organizing input text with attention mask adjustments. Results on English and Chinese GEC benchmarks demonstrate that our proposed model surpasses previous detection-correction models and is comparable to state-of-the-art (SOTA) models. To further explore the potential of applying the detection-correction structure to LLMs, the detection and correction phases are separated, termed DeGLM and CoGLM respectively. Our proposed single system, comprising a small detection model and an LLM corrector, outperforms other Seq2Seq LLMs. In summary, our primary contributions are:

- A novel GEC model, DeCoGLM, which incorporates a detection-correction structure based on the GLM.
- The design of a multi-task training method that integrates detection and correction within a single model.
- The exploration of using LLMs for GEC, which involves deploying large error correction models with the support of small detection models.

2 Related Work

2.1 Sequence-to-Sequence GEC

Seq2Seq models (Lewis et al., 2019; Raffel et al., 2020) have demonstrated high performance in GEC (Junczys-Dowmunt et al., 2018; Choe et al., 2019; Zhao et al., 2019; Katsumata and Komachi, 2020). Techniques such as data synthesis (Stahlberg and Kumar, 2021; Grundkiewicz et al., 2019), training schedule (Lichtarge et al., 2020; Bout et al., 2023), and decode reranking methods (Kaneko et al., 2019; Zhang et al., 2023; Zhou et al., 2023) have been incorporated into previous Seq2Seq GEC models. SOTA model architectures typically supplement Seq2Seq models with additional information (Li et al., 2023a; Zhang et al., 2022b; Fang et al., 2023a). However, a significant drawback of Seq2Seq GEC models is the inference cost, as these models generate tokens sequentially and waste time copying source tokens (Sun et al., 2021).

As the latest Seq2Seq models, LLMs have emerged as a new paradigm for natural language processing (NLP) tasks following the introduction of GPT-3 and ChatGPT (Brown et al., 2020). Nevertheless, recent studies have shown that LLMs underperform current SOTA models on both English and Chinese GEC benchmarks (Coyne et al., 2023; Loem et al., 2023; Qu and Wu, 2023; Li et al., 2023b). Existing datasets and evaluation methods (Bryant et al., 2017) favor minimum edits as the rule for correction. However, GPT models often produce over-corrected sentences with unnecessary edits (Fang et al., 2023b; Coyne et al., 2023). In contrast to the Seq2Seq GEC methods that directly perform overall generation, our work only focuses on localized error correction, which not only saves inference time but also mitigates the over-correction phenomena in LLMs.

2.2 Sequence-to-Edit GEC

Seq2Edit methods generate edit operations for ungrammatical sentences (Stahlberg and Kumar, 2020). For instance, LaserTagger (Malmi et al., 2019) predicts token-level edit operations, which has been adopted in subsequent methods like PIE and GECToR (Awasthi et al., 2019; Omelianchuk et al., 2020). As a representative model, GECToR predicts four classes of edits and grammatical transformations, achieving high-precision results. Lai et al. (2022) further enhances it by addressing its deficiencies in multi-round correction. However, Seq2Edit methods necessitate intricate designs for edits, which are not language-agnostic. In contrast, our proposed model retains a limited set of language-agnostic edit operations and can flexibly conduct edits by autoregressive generation.

2.3 Detection-Correction GEC

The GEC task can be divided into two processes: detection and correction (Rei and Yannakoudakis, 2016; Bell et al., 2019). Prior research incorporates detection results as supplementary information for Seq2Seq correction models (Kaneko et al., 2020; Yuan et al., 2021b; Li et al., 2023a). The methods proposed by Mallinson et al. (2020) and Yakovlev et al. (2023) employ the Masked Language Model (MLM) (Devlin et al., 2018) to obtain corrections, which are constrained by mask number. Chen et al. (2020) introduces error span detection and correction to address the GEC problem, which allows for flexible corrections while maximizing time efficiency. Building on this, we further integrate the detection and correction tasks into a single GLM model, enabling mutual benefits between the two tasks, which is not achieved by previous works.

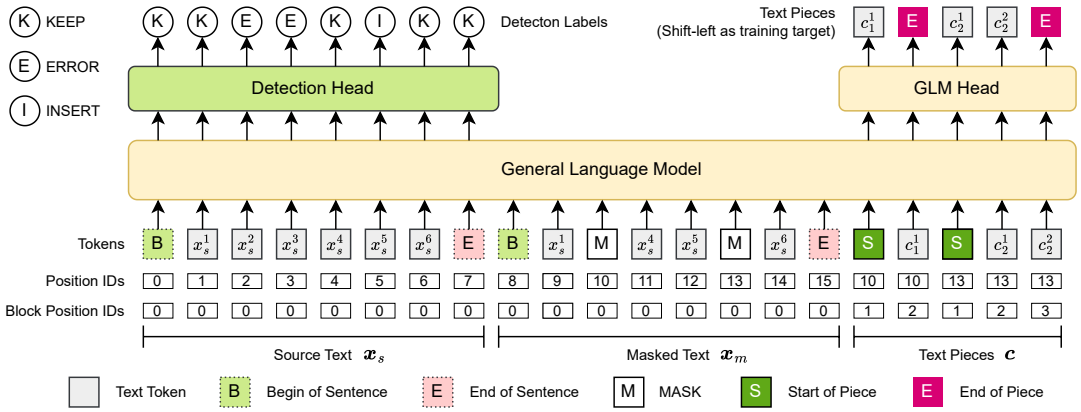

Figure 2: The proposed **detection-correction structure** based on GLM. The example shown above has a source text $\boldsymbol{x}_s = x_s^1 x_s^2 x_s^3 x_s^4 x_s^5 x_s^6$ and the target text is $\boldsymbol{y} = x_s^1 c_1^1 x_s^4 x_s^5 c_2^1 c_2^2 x_s^6$. Consistent with GLM, the position IDs and block position IDs are utilized for marking the original positions of text pieces and the inner order of tokens.

3 Methods

Our proposed model leverages the design of the GLM. Given a sentence with MASK tokens, GLM utilizes autoregressive blank infilling (Du et al., 2022) to generate a corresponding segment for each MASK position. These segments are termed as **text pieces**. This section describes how GLM is utilized to integrate detection and correction into a single model, as depicted in Figure 2. Additionally, the design of multi-task training is also outlined here.

3.1 Error Detection

Drawing from the four edit classes by Omelianchuk et al. (2020), we utilize token-level detection labels that do not include any specific word or grammar. Given that the mask-infilling process can generate empty text pieces, the REPLACE and DELETE operations are consolidated into the ERROR label. Consequently, the detection labels comprise KEEP (K), ERROR (E), and INSERT (I). Given the tokens of **source text** as:

$$\boldsymbol{x}_s = x_s^1 x_s^2 \dots x_s^n \tag{1}$$

, the objective of error detection is to predict **detection labels** derived by the alignment between the source text and the target text (correct text):

$$\boldsymbol{d} = d_1 d_2 \dots d_n, d_i \in L = \{K, E, I\} \tag{2}$$

Detection Model The proposed model begins by extracting the representations of the source text tokens by GLM as Equation 3. The final detection label predictions are generated through a detection head, implemented by a feed-forward network FN and softmax function, as shown in Equation 4:

$$\boldsymbol{h}_s = h_s^1 h_s^2 \dots h_s^n = \mathrm{GLM}\left(\boldsymbol{x}_s\right) \tag{3}$$

$$p\left(\hat{d}_i = l | \boldsymbol{x}_s\right) = \mathrm{Softmax}(\mathrm{FN}\left(h_s^i\right)), l \in L \tag{4}$$

Fault-tolerant Template The source text $\boldsymbol{x}_s$ is transformed into masked text $\boldsymbol{x}_m$ based on the detection labels using the following template rules. Each continuous interval containing only ERROR labels is replaced with a MASK token. For each position of INSERT, a MASK token is inserted. The form of **masked text** is shown in Equation 5:

$$\boldsymbol{x}_m = \boldsymbol{x}_{s_1} m_1 \boldsymbol{x}_{s_2} m_2 \dots m_k \boldsymbol{x}_{s_{k+1}}, \tag{5}$$

where m_i is the i-th MASK token introduced in $\boldsymbol{x}_s$, and $\boldsymbol{x}_{s_i}$ denotes the i-th correct subinterval of source text. If all the labels are KEEPs, the source text is directly output as the corrected result. Despite potential inaccuracies in detections, our model can tolerate a certain degree of false positives. In the instance where the correct token is identified as ERROR or INSERT, the corrector can mitigate such errors by either restoring the original text piece or generating an empty text piece.

Aggressive Detection Utilizing the fault-tolerant template enables more aggressive detection, emphasizing the recall of ERROR and INSERT. Focal Loss (Lin et al., 2020) is used as the loss function to tackle the issue of imbalanced classification because the majority of tokens correspond to KEEP labels. The training objective for error detection is given by Equation 6:

$$\ell_D = -\boldsymbol{\alpha}_D \left(1 - p_\theta\left(\boldsymbol{d}|\boldsymbol{x}_s\right)\right)^\gamma \log\left(p_\theta\left(\boldsymbol{d}|\boldsymbol{x}_s\right)\right) \tag{6}$$

where θ represents the model parameters and γ is a hyper-parameter set to 2. $\boldsymbol{\alpha}_D$ denotes the corresponding weight factors for detection labels. To strengthen aggressive error detection, α_K for the

KEEP category is set to 1, while $\alpha_{EI} = 2$ is set for the ERROR and INSERT categories.

3.2 Localized Error Correction

In the training data, detection labels are derived from the alignment of sequences between the source text $\boldsymbol{x}_s$ and the target text $\boldsymbol{y}$. The corresponding masked text $\boldsymbol{x}_m$ can be formulated in Equation 5 with $\boldsymbol{x}_{s_i}$ representing the i-th aligned segment. For each unaligned position replaced with m_i, the correct text piece is denoted as $\boldsymbol{c}_i$. Consequently, the target text can be represented as:

$$\boldsymbol{y} = \boldsymbol{x}_{s_1}\boldsymbol{c}_1\boldsymbol{x}_{s_2}\boldsymbol{c}_2\ldots\boldsymbol{c}_k\boldsymbol{x}_{s_{k+1}} \tag{7}$$

Leveraging the GLM pretrained by autoregressive blank infilling task, we fine-tune the GLMs for localized error correction. The probability distribution prediction for the j-th token in the i-th text piece $\boldsymbol{c}_i$ is given in Equation 8:

$$\begin{aligned} p\left(\hat{c_{i,j}} = w|\boldsymbol{x}_s, \boldsymbol{x}_m, \boldsymbol{c}_{<i}, \boldsymbol{c}_i^{<j}\right) = \\ \text{GLMH}\left(\boldsymbol{x}_s, \boldsymbol{x}_m, \boldsymbol{c}_{<i}, \boldsymbol{c}_i^{<j}\right), w \in V \end{aligned} \tag{8}$$

where GLMH denotes the GLM model with its original token prediction head, w is any token in the vocabulary, and $\boldsymbol{c}_i^{<j}$ refers to all tokens with index $< j$ in text piece $\boldsymbol{c}_i$.

3.3 Multi-Task Organization

Multi-task Learning. The cross-entropy loss function, shown in Equation 9, is used as the training objective for error correction task:

$$\ell_C = -\sum_{i,j}\log\left(p_\theta\left(c_{i,j}|\boldsymbol{x}_s, \boldsymbol{x}_m, \boldsymbol{c}_{<i}, \boldsymbol{c}_i^{<j}\right)\right) \tag{9}$$

For multi-task learning, we utilize a weighted loss function to enable the model to concurrently acquire error detection and correction capabilities. The training objective for this DeCoGLM model is to minimize the loss function given by:

$$\ell = \bar{\ell_C} + w_D\bar{\ell_D} \tag{10}$$

where $\bar{\ell_C}$ and $\bar{\ell_D}$ are the token-level averages of ℓ_C and ℓ_D respectively. The detection loss weight w_D is set to 10 to balance the scales of the two losses. For the impact of the loss weights on the model's performance, please refer to Section 5.1.

Attention Mask To unify the two tasks into a single model, source text $\boldsymbol{x}_s$, masked text $\boldsymbol{x}_m$, and text pieces $\boldsymbol{c}$ are concurrently fed into the GLM model. The prediction of detection labels is conditioned on $\boldsymbol{x}_s$, while the autoregressive text prediction relies on $\boldsymbol{x}_s$, $\boldsymbol{x}_m$, and all previously generated text pieces. Therefore, the attention from $\boldsymbol{x}_m$ to $\boldsymbol{x}_s$ is eliminated to prevent detection from using $\boldsymbol{x}_m$ and $\boldsymbol{c}$, with other part adhering to the original GLM attention mask. This is depicted in Figure 3.

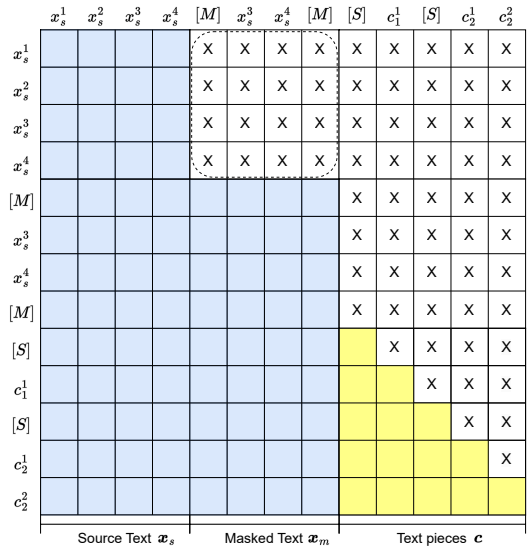

Figure 3: **Attention Mask Example.** The source text is $\boldsymbol{x}_s = x_s^1x_s^2x_s^3x_s^4$, and the target text is $\boldsymbol{y} = c_1^1x_s^3x_s^4c_2^1c_2^2$. The region enclosed by dashed lines indicates the attention removed compared to the original GLM.

Two Stage Supervised Fine-tuning Given that error detection is not infallible, the input during the correction phase may contain inaccuracies, with a distribution deviation from the training samples constructed with right detection labels. This issue is also observed in other detection-correction works (Chen et al., 2020; Li et al., 2023a). To address this, we add a second supervised fine-tuning stage (SFT2), which employs a **detection-enhanced** approach: initially, all detection results on the training set are obtained using the model trained with data constructed by perfect detection (SFT1). Then, new training data is generated by augmenting the original labels with the fault detection results, leading to a secondary training of the SFT1 model. Examples of the two-stage training samples are provided in Table 6 in the appendix.

3.4 Separate Models

The detection and correction phases can be implemented using two separate GLMs, named DeGLM and CoGLM respectively in this paper. Their training objectives are defined by Equation 6 and 9, respectively. This decomposition facilitates the

customization of distinct models for the detection and correction phases. However, in scenarios with limited computational resources, the Parameter-Efficient Fine-Tuning (PEFT) (Fu et al., 2023) is ineffective for DeCoGLM due to the significant disparities between the sequence labeling task of the error detection module and the mask-infilling pretraining task of GLM. To apply our approach to LLMs, we propose training a large version of CoGLM using the detection-enhanced method, similar to the second stage fine-tuning discussed in Section 3.3.

3.5 Detection Control

During inference, the model needs to predict detection labels for the source text first, then transform it into masked text. Subsequently, both of them are input together for generating text pieces. This decoupling allows us to regulate the correction process using the probabilities of the three detection labels, thereby harnessing the model's potential to enhance benchmark performance. Three control modes are designed:

KEEP Threshold (δ): Any prediction with KEEP probability exceeding δ is directly set to KEEP.

ERROR Lower Bound (ϕ_e): Any ERROR probability prediction falling below ϕ_e is directly set to 0, thereby precluding the prediction of ERROR when $p_e < \phi_e$.

INSERT Lower Bound (ϕ_i): Any INSERT probability prediction below ϕ_i is directly set to 0, precluding the prediction of INSERT when $p_i < \phi_i$.

The three inference hyper-parameters can be determined using a greedy grid search based on the metrics on the validation set. We discuss them in Section 5.4.

4 Experiments

4.1 Datasets and Evaluation

For the English GEC task, we evaluate the performance on the CoNLL-14 test set (Ng et al., 2014) using the M^2 Scorer (Dahlmeier and Ng, 2012), and on the BEA-19 test set (Bryant et al., 2019) using the ERRANT scorer (Bryant et al., 2017). The model is pretrained on synthetic dataset C4-200M (Stahlberg and Kumar, 2021) and fine-tuned on the cleaned Lang8 dataset (CLang8) (Rothe et al., 2021). For the large version of CoGLM model, we utilize smaller datasets including FCE (Yannakoudakis et al., 2011), NUCLE (Dahlmeier et al., 2013), and W&I+LOCNESS (Bryant et al., 2019) for fine-tuning, following Zhou et al. (2023). The BEA-19 dev set is used for model selection.

For the Chinese GEC task, we synthesize pretraining data from the People's Daily corpus[1] using rule-based insertion, replacement, and deletion. The models are fine-tuned on the Chinese Lang8 dataset (Zhao et al., 2018) and the HSK dataset, following Zhang et al. (2022a), and on the FCGEC training set, respectively. The models are evaluated on MuCGEC and FCGEC test sets using ChERRANT (Zhang et al., 2022a; Xu et al., 2022). Further details are provided in Appendix A.

4.2 Model Settings

Proposed Models The open-source GLMs are utilized as the backbones for both DeCoGLM and separate models. The detection head comprises a feed-forward network with a single hidden layer, the dimension of which matches that of the GLM hidden state. The English base model employs `glm-roberta-large`, while `glm-large-chinese` is used as the Chinese base model. The large CoGLM models for error correction, denoted as CoGLM (10B), uses `glm-10b` and `glm-10b-chinese` as backbones. Due to the restriction of computational resources, large models are fine-tuned on the relatively small fine-tuning dataset mentioned in Section 4.1 by LoRA (Hu et al., 2021), without datasets for pretraining. Refer to Appendix B.2 for detailed configurations.

Comparison with Previous Works In the main experiment, we present the results of single systems trained on parallel data without reranker. GECToR (Omelianchuk et al., 2020) represents the Seq2Edit models, while BART and T5 (Lewis et al., 2019; Raffel et al., 2020) are SOTA backbones of Seq2Seq GEC methods. SynGEC (Zhang et al., 2022b) incorporates syntactic information into the BART model. The performance of GECToR and BART model on the Chinese dataset is the reproduced result under our data configuration, and the results for BART on the English dataset are reported by Zhang et al. (2022b). We also present the results of four models involving the detection-correction process. SpanDC (Chen et al., 2020) comprises a span detector and a generator. Multi-Encoder (Yuan et al., 2021a) encodes error categories as auxiliary information. GEC-DePend (Yakovlev et al., 2023) integrates error detection with correction by the MLM. TemplateGEC (Li

[1] https://github.com/shibing624/pycorrector

		English						Chinese					
		CoNLL-14 *test*			BEA-19 *test*			MuCGEC *test*			FCGEC *test*		
Single System	**Parameters**	**P**	**R**	**$F_{0.5}$**	**P**	**R**	**$F_{0.5}$**	**P**	**R**	**$F_{0.5}$**	**P**	**R**	**$F_{0.5}$**
Primary Results													
GECToR	110M	**77.5**	40.1	65.3	**79.2**	53.9	72.4	46.72	27.14	40.83	46.11	34.35	43.16
BART	400M	73.6	48.6	66.7	74.0	64.9	72.0	41.90	29.48	38.64	38.38	37.62	38.23
T5	770M	-	-	66.1	-	-	72.1	-	-	-	-	-	-
SynGEC	110M+400M	74.7	49.0	67.6	75.1	**65.5**	72.9	**54.69**	29.10	**46.51**	-	-	-
SpanDC	125M+209M	72.6	37.2	61.0	70.4	55.9	66.9	-	-	-	-	-	-
Multi-Encoder	110M+107M	71.3	44.3	63.5	73.3	61.5	70.6	-	-	-	-	-	-
GEC-DePenD	253M	73.2	37.8	61.6	72.9	53.2	67.9	-	-	-	-	-	-
TemplateGEC	125M+770M	74.8	**50.0**	**68.1**	76.8	64.8	74.1	-	-	-	-	-	-
DeGLM-CoGLM	335M+335M	75.1	49.0	67.8	76.4	63.4	73.4	47.22	30.08	42.39	52.95	**39.20**	49.48
DeCoGLM	335M	75.1	49.4	68.0	77.4	64.6	**74.4**	45.01	**31.77**	41.55	**55.75**	37.91	**50.96**
Resource-restricted LLMs													
ChatGLM2	6B	61.72	45.58	57.64	56.89	58.73	57.25	31.35	21.39	28.68	44.30	17.08	33.59
ChatGLM3	6B	60.63	47.50	57.46	59.48	60.37	59.65	30.62	21.60	28.26	41.06	19.93	33.88
LLaMA2/Baichuan	7B	67.24	51.84	63.47	66.16	66.12	66.15	36.47	25.18	33.47	51.83	24.08	42.12
LLaMA2/Baichuan	13B	68.43	**55.30**	65.33	69.46	**69.28**	69.42	37.91	26.90	35.04	**56.65**	27.11	46.52
DeGLM-CoGLM	335M+10B	**70.58**	52.65	**66.08**	**72.80**	67.57	**71.69**	**47.48**	**29.92**	**42.49**	56.09	**38.02**	**51.22**
GPT-4 Zeroshot													
ZeroShot	-	59.64	**58.32**	59.37	55.69	**70.44**	58.13	**36.36**	27.71	**34.22**	18.83	4.08	10.93
+DeGLM	-	**66.40**	54.81	**63.70**	**64.92**	69.42	**65.78**	32.68	**30.90**	32.31	**25.60**	**16.98**	**23.24**

Table 1: Results on English and Chinese GEC benchmarks. The parameter counts of the backbones of each system are shown in the second column. Under restricted resource, LLMs are fine-tuned using smaller datasets by LoRA. The highest metric is indicated in bold, while the second highest metric value is underlined.

et al., 2023a) uses the output of the GECToR model as supplementary information for Seq2Seq models.

Comparison with LLMs For the LLMs treating GEC as a Seq2Seq task, we fine-tune ChatGLM2, ChatGLM3 (Du et al., 2022), and Llama2 (Touvron et al., 2023) with LoRA. As Llama2 is not optimized for Chinese, the results on the Chinese dataset are obtained using the Baichuan (Yang et al., 2023) models.

GPT-4 We report the zero-shot performance of GPT-4 on four datasets with prompting. We attempt to incorporate detection results in the form of masked text into the prompt of GPT-4, aiming to enhance the performance on GEC tasks.

4.3 Main Results

Table 1 presents the main results. According to the last two rows of primary results, the integrated detection-correction model outperforms the separate models in most cases in terms of the $F_{0.5}$ metric, despite having only half the parameter count. This suggests that the designed multi-task learning mutually reinforces detection and correction. DeCoGLM achieves the highest or second-highest $F_{0.5}$ performance on three datasets, demonstrating comparable performance to SOTA GEC models. Considering the model parameter counts, our model outperforms all previous works with the detection-correction process, indicating that the well-designed detection-correction structure can achieve SOTA level in GEC, a field typically dominated by Seq2Seq models. These results also underscore the potential of GLM in GEC field.

Despite limitations of data quantity and fine-tuning methods, fine-tuning LLMs with over 10B parameters yields results approaching SOTA level, suggesting that LLMs can reduce the need for extensive supervised data for fine-tuning. The strategy of small detection models assisting large models in localized correction yields improved performance across all datasets, primarily due to higher precision. This suggests that the model reduces over-correction at the expense of a certain level of recall. On the English dataset, GPT-4 exhibits a similar trend when incorporated with detection results, indicating that integrating detections can stably improve the GEC capability of LLMs, thus presenting a promising future direction for GEC.

5 Analysis

5.1 Weights of Multi-Task Training

To establish two weights that significantly impact the training objective: the detection loss weight w_D in Equation 10, and the ERROR and INSERT loss weight α_{EI} in Equation 6, we conduct preliminary experiments, which include only the two stages of fine-tuning. The obtained results are presented in Table 3. Based on a preliminary observation on the loss scale, we initially set $w_D = 10$ and

BackBone	Pretrained	SFT1	SFT2	Ctrl	CoNLL-14 *test* P	CoNLL-14 *test* R	CoNLL-14 *test* $F_{0.5}$	BEA-19 *test* P	BEA-19 *test* R	BEA-19 *test* $F_{0.5}$
GLM-Roberta	Yes	✓	✓	✓	75.07	49.40	**68.00**	**77.36**	64.63	**74.43**
GLM-Roberta	Yes	✓	✓	×	70.47	54.96	66.70	72.75	69.28	72.03
GLM-Roberta	Yes	✓	×	✓	**75.27**	48.24	67.69	76.55	62.34	73.21
GLM-Roberta	Yes	✓	×	×	68.38	**57.35**	65.84	69.00	**71.02**	69.39
GLM-Roberta	Yes	×	×	×	54.04	45.99	52.21	45.12	58.60	47.30
GLM-Roberta	No	✓	✓	✓	**72.78**	46.42	**65.36**	**75.54**	59.87	**71.78**
GLM-Roberta	No	✓	✓	×	69.25	51.26	64.71	72.33	65.46	70.85
GLM-Roberta	No	✓	×	✓	68.25	49.33	63.39	69.66	61.94	67.97
GLM-Roberta	No	✓	×	×	63.92	**52.46**	61.25	66.27	**66.01**	66.21
BART-large	No	✓	✓	✓	**69.53**	45.62	**62.93**	**72.01**	57.84	**68.64**
BART-large	No	✓	✓	×	66.39	49.80	62.24	69.25	63.28	67.97
BART-large	No	✓	×	✓	67.54	43.66	60.88	68.08	55.14	65.03
BART-large	No	✓	×	×	62.75	**50.40**	59.81	64.67	**63.62**	64.46

Table 2: Ablation study results. The "Ctrl" denotes the proposed detection control.

w_D	α_{EI}	$F_{0.5}$ on dev set BEA-19	MuCGEC	FCGEC
20	**2**	60.30	34.45	40.57
10	-	60.09	35.17	41.52
	1	59.93	34.25	**42.89**
	2	**60.81**	35.09	42.49
	3	60.29	**35.82**	40.72
	4	60.12	35.03	41.49
5	**2**	60.60	34.53	42.10
1	**2**	59.64	33.23	36.72

Table 3: The preliminary experimental results of different loss weights. w_D and α_{EI} is defined in Section 3.3 and 3.1. The "-" value of α_{EI} represents the usage of cross-entropy other than Focal Loss.

K	E	I	D	CoNLL-14 *test* P	CoNLL-14 *test* R	CoNLL-14 *test* $F_{0.5}$	BEA-19 *test* P	BEA-19 *test* R	BEA-19 *test* $F_{0.5}$
✓	✓			**69.67**	50.91	**64.89**	72.18	65.14	70.65
✓	✓	✓		69.25	**51.26**	64.71	**72.33**	**65.46**	**70.85**
✓	✓	✓	✓	68.48	49.95	63.75	71.23	64.48	69.77

Table 4: Results under different detection label sets.

explore experimental results under varying α_{EI}. The outcomes suggest that the Focal Loss along with moderately increasing α_{EI} to achieve aggressive detection introduced in Section 3.3 is effective. After setting $\alpha_{EI} = 2$, we conducted additional experiments with different w_D. The overall experimental results indicate that $\alpha_{EI} = 2$ and $w_D = 10$ constitute a suitable setup.

5.2 Detection Label Set

In the design outlined in Section 3.1, ERROR includes both replacement and deletion, as the deletion can be considered as replacing with zero-length text. The results for this design are shown in the second row of Table 4. INSERT can also be further merged into the ERROR label. This can be achieved by considering the INSERT operation as replacing the token x_i at the insertion position with $x_i \boldsymbol{c_j}$, where $\boldsymbol{c_j}$ represents tokens to be inserted. The results corresponding to this approach are shown in the first row of Table 4. Additionally, we demonstrate the results of applying four detection labels (KEEP, ERROR, INSERT, DELETE) in the last row. Overall, our designed three-label scheme performs relatively better, as the insertion operation in the two-label mode requires disrupting the correct part of the source text, and encountering DELETE in the four-label mode will lead to direct deletion, which makes the model unable to recover from faults in the error correction phase.

5.3 Ablation Study

To explore the effectiveness of various components in the designed detection-correction model, we conduct an ablation study focusing on synthetic data, backbone, two-stage fine-tuning, and detection control. The results are shown in Table 2.

Effectiveness of synthetic data In the proposed model, both the English and Chinese models undergo pretraining with a large-scale synthetic dataset of GEC. A comparison between the top and middle rows of Table 2 reveals that pretraining indeed provides a stable improvement in model performance, although the data used is not from real scenarios.

Effectiveness of GLM backbone The detection-correction structure can also be implemented in Seq2Seq models. We applied the proposed method to the BART model and conducted experiments. An additional detection head is integrated into the BART encoder, while the decoder generates text pieces for localized error correction. The experimental results, depicted in the bottom rows of Table 2, consistently demonstrate superior performance when employing GLM as the backbone compared

to using BART. This can be attributed, in part, to the consistency between the original pretraining task of GLM and the training objective of the correction task, as defined in Equation 9. However, the pretraining pattern of BART differs. Additionally, the separation of BART's encoder and decoder into two distinct modules may not effectively foster the mutual enhancement of detection and correction abilities in multi-task learning.

Effectiveness of Two Stage Fine-tuning As described in Section 3.3, two fine-tuning stages differ in the training data: SFT1 constructs training samples using only ground-truth detection labels, while SFT2 utilizes both ground-truth detection labels and the detection results from the model trained in the first stage. As evident from the comparison in Table 2, SFT1 significantly improves the model's performance than the model pretrained on the synthetic dataset. Comparing the results exclusively differing in SFT2 in Table 2, it is observed that SFT2 consistently enhances $F_{0.5}$, primarily attributed to the improvement in precision while maintaining recall relatively constant. This validates the effectiveness of the two-stage supervised fine-tuning design.

Detection Control From Table 2, it is evident that, under the scenario of employing the same trained model, setting three hyper-parameters for the detection phase also enhances the $F_{0.5}$ performance. This approach primarily aims at improving precision. However, upon closer inspection, it is noticeable that this technique results in a more substantial reduction in recall compared to the second-stage fine-tuning. For all GLM models incorporating detection control, the recall on the CoNLL-14 test set is consistently below 50%, and the recall on the BEA-19 test set is consistently below 65%. Thus, the effectiveness of detection control stems more from the trade-off between precision and recall, as discussed in the next section.

5.4 Precision-Recall Trade off

Adjusting the threshold for KEEP prediction probability (δ) and the probability lower bounds for ERROR and INSERT predictions (ϕ_e, ϕ_i) defined in Section 3.5 allows for further adjustment of precision and recall, resulting in improved $F_{0.5}$ scores. We performed a parameter search on the validation set to identify configurations maximizing $F_{0.5}$, and the results are depicted in Figure 4.

Without setting ϕ_e and ϕ_i, $\delta = 0.38$ achieved the highest $F_{0.5}$ of 63.2 on BEA-19 dev set. Then, we

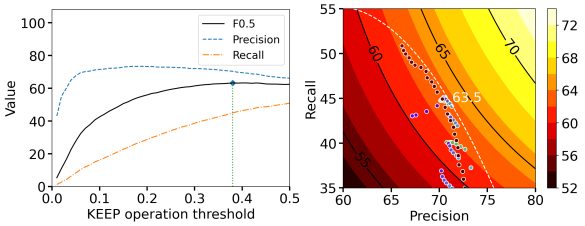

Figure 4: Results of detection control on BEA-19 dev set. The heat value represents the value of $F_{0.5}$.

fix $\delta = 0.38$ and perform a grid search for ϕ_e, ϕ_i. All results are presented as points in the right plot of Figure 4, and nearly all points are located within the region enclosed by the dashed line in the bottom-left. The dashed line represents the boundary of the model's capability, and the intersection point with the $F_{0.5}$ contour line represents the optimal performance attainable by the model. The point with the highest $F_{0.5} = 63.5$ is the one closest to the intersection point, with $\phi_e = 0.5$ and $\phi_i = 0.6$. Under this parameter configuration, the model achieved an $F_{0.5}$ value of 74.43 on the BEA-19 test set, as shown in Table 1. The detection control offers such a straightforward implementation of the precision-recall trade-off.

6 Conclusion

We introduce a novel language-agnostic detection-correction structure via GLM for the GEC task. The structure employs a three-label error detection pattern and uses Focal Loss for aggressive detection. The correction phase leverages the mask-infilling capability of GLM to generate correct text pieces. A multi-task learning approach is designed to integrate both functionalities within the same model, optimized using a weighted loss function. Experimental results show proposed model DeCoGLM outperforms previous detection-correction structures and achieves $F_{0.5}$ scores comparable to SOTA on English and Chinese GEC benchmarks. The effectiveness of the detection-correction structure is further validated by applying it to open-source LLMs and GPT-4, indicating that incorporating error detection information improves the performance of LLMs on GEC datasets by reducing over-correction. Ablation studies confirm the efficacy of our model design and the ability to trade off precision and recall can be realized by detection control. We aim for this work to further guide research in the GEC field within the detection-correction paradigm.

Limitations

Incremental methods proven effective on Seq2Seq models, such as incorporating syntactic information (Zhang et al., 2022b), refining training data (Mita et al., 2020), and employing additional models for reranking during the generation phase (Zhang et al., 2023; Zhou et al., 2023), are not implemented in this work. The main objective of this paper is to propose a novel GEC architecture, with these additional tricks serving as potential avenues for future extensions. Furthermore, due to resource restrictions, we are unable to apply our integrated detection-correction structure to LLMs. This is because the sequence labeling task differs from the generative tasks that LLMs are designed to perform, necessitating full-parameter fine-tuning to integrate the two tasks. Additionally, in our investigation of LLMs as correction models, models with parameters exceeding 13B are not utilized. The absence of full-parameter fine-tuning on LLMs and experiments with larger models due to resource constraints leaves room for further exploration of the application of the detection-correction paradigm on LLMs.

Ethics Statement

The datasets and models we used are publicly available and utilized only for research purposes. The datasets do not contain any information that names or uniquely identifies individual people or offensive content. LLMs are utilized in our experiments, consistent with their intended use in natural language processing tasks. The models we designed will be published and intended for academic research in the field of grammatical error correction, in accordance with the original access conditions of the models used.

The detection-correction structure we designed limits the model to making only localized modifications to the text, preventing it from generating text without constraints, thereby significantly reducing the potential risks associated with the model. However, It is worth noting that the modifications made by the designed model may alter certain facts in the text, leading to hallucination, especially when modifications occur in named entities.

ChatGPT is utilized as the AI Assistant to polish the paper writing.

References

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4260–4270, Hong Kong, China. Association for Computational Linguistics.

Samuel Bell, Helen Yannakoudakis, and Marek Rei. 2019. Context is key: Grammatical error detection with contextual word representations. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 103–115, Florence, Italy. Association for Computational Linguistics.

Andrey Bout, Alexander Podolskiy, Sergey Nikolenko, and Irina Piontkovskaya. 2023. Efficient grammatical error correction via multi-task training and optimized training schedule. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5800–5816, Singapore. Association for Computational Linguistics.

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.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Andrew Caines, Luca Benedetto, Shiva Taslimipoor, Christopher Davis, Yuan Gao, Oeistein Andersen, Zheng Yuan, Mark Elliott, Russell Moore, Christopher Bryant, et al. 2023. On the application of large language models for language teaching and assessment technology. *arXiv preprint arXiv:2307.08393*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.* Just Accepted.

Mengyun Chen, Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2020. Improving the efficiency of grammatical error correction with erroneous span detection and correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7162–7169, Online. Association for Computational Linguistics.

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, Florence, Italy. Association for Computational Linguistics.

Steven Coyne, Keisuke Sakaguchi, Diana Galvan-Sosa, Michael Zock, and Kentaro Inui. 2023. Analyzing the performance of gpt-3.5 and gpt-4 in grammatical error correction. *arXiv preprint arXiv:2303.14342*.

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, Montréal, Canada. Association for Computational Linguistics.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, Atlanta, Georgia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, Dublin, Ireland. Association for Computational Linguistics.

Tao Fang, Jinpeng Hu, Derek F. Wong, Xiang Wan, Lidia S. Chao, and Tsung-Hui Chang. 2023a. Improving grammatical error correction with multimodal feature integration. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9328–9344, Toronto, Canada. Association for Computational Linguistics.

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

Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. 2023. On the effectiveness of parameter-efficient fine-tuning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11):12799–12807.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065, Melbourne, Australia. Association for Computational Linguistics.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 252–263, Florence, Italy. Association for Computational Linguistics.

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*.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 595–606, New Orleans, Louisiana. Association for Computational Linguistics.

Masahiro Kaneko, Kengo Hotate, Satoru Katsumata, and Mamoru Komachi. 2019. TMU transformer system using BERT for re-ranking at BEA 2019 grammatical error correction on restricted track. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 207–212, Florence, Italy. Association for Computational Linguistics.

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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254, Online. Association for Computational Linguistics.

Masahiro Kaneko, Sho Takase, Ayana Niwa, and Naoaki Okazaki. 2022. Interpretability for language learners using example-based grammatical error correction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7176–7187, Dublin, Ireland. Association for Computational Linguistics.

Anisia Katinskaia and Roman Yangarber. 2021. Assessing grammatical correctness in language learning. In

Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications, pages 135–146, Online. Association for Computational Linguistics.

Satoru Katsumata and Mamoru Komachi. 2020. Stronger baselines for grammatical error correction using a pretrained encoder-decoder model. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 827–832, Suzhou, China. Association for Computational Linguistics.

Shaopeng Lai, Qingyu Zhou, Jiali Zeng, Zhongli Li, Chao Li, Yunbo Cao, and Jinsong Su. 2022. Type-driven multi-turn corrections for grammatical error correction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3225–3236, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):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, Toronto, Canada. Association for Computational Linguistics.

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*.

Junwei Liao, Sefik Eskimez, Liyang Lu, Yu Shi, Ming Gong, Linjun Shou, Hong Qu, and Michael Zeng. 2023. Improving readability for automatic speech recognition transcription. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 22(5).

Jared Lichtarge, Chris Alberti, and Shankar Kumar. 2020. Data weighted training strategies for grammatical error correction. *Transactions of the Association for Computational Linguistics*, 8:634–646.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327.

Mengsay Loem, Masahiro Kaneko, Sho Takase, and Naoaki Okazaki. 2023. Exploring effectiveness of gpt-3 in grammatical error correction: A study on performance and controllability in prompt-based methods. *arXiv preprint arXiv:2305.18156*.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. FELIX: Flexible text editing through tagging and insertion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1244–1255, Online. Association for Computational Linguistics.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065, Hong Kong, China. Association for Computational Linguistics.

Masato Mita, Shun Kiyono, Masahiro Kaneko, Jun Suzuki, and Kentaro Inui. 2020. A self-refinement strategy for noise reduction in grammatical error correction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 267–280, Online. Association for Computational Linguistics.

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, Baltimore, Maryland. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem 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*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

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*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the

limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Marek Rei and Helen Yannakoudakis. 2016. Compositional sequence labeling models for error detection in learner writing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1181–1191, Berlin, Germany. Association for Computational Linguistics.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error 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 2: Short Papers)*, pages 702–707, Online. Association for Computational Linguistics.

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, Online. Association for Computational Linguistics.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47, Online. Association for Computational Linguistics.

Hanbo Sun, Jian Gao, Xiaomin Wu, Anjie Fang, Cheng Cao, and Zheng Du. 2023. Htec: Human transcription error correction. *arXiv preprint arXiv:2309.10089*.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. 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 5937–5947, Online. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Yu Wang, Yuelin Wang, Kai Dang, Jie Liu, and Zhuo Liu. 2021. A comprehensive survey of grammatical error correction. *ACM Trans. Intell. Syst. Technol.*, 12(5).

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, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Konstantin Yakovlev, Alexander Podolskiy, Andrey Bout, Sergey Nikolenko, and Irina Piontkovskaya. 2023. GEC-DePenD: Non-autoregressive grammatical error correction with decoupled permutation and decoding. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1546–1558, Toronto, Canada. Association for Computational Linguistics.

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*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

Zheng Yuan, Shiva Taslimipoor, Christopher Davis, and Christopher Bryant. 2021a. Multi-class grammatical error detection for correction: A tale of two systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8722–8736, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zheng Yuan, Shiva Taslimipoor, Christopher Davis, and Christopher Bryant. 2021b. Multi-class grammatical error detection for correction: A tale of two systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8722–8736, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Ying Zhang, Hidetaka Kamigaito, and Manabu Okumura. 2023. Bidirectional transformer reranker for grammatical error correction. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3801–3825, Toronto, Canada. Association for Computational Linguistics.

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, Seattle, United States. Association for Computational Linguistics.

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, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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 American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 156–165, Minneapolis, Minnesota. Association for Computational Linguistics.

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*, pages 439–445, Cham. Springer International Publishing.

Houquan Zhou, Yumeng Liu, Zhenghua Li, Min Zhang, Bo Zhang, Chen Li, Ji Zhang, and Fei Huang. 2023. Improving Seq2Seq grammatical error correction via decoding interventions. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7393–7405, Singapore. Association for Computational Linguistics.

A Dataset

A.1 Dataset Statistics

Dataset	#Sentences	Usage	As training data of
C4-200M	183,894,319	Pretraining	DeCoGLM, DeGLM, CoGLM
Synthetic-CH	33,166,047	Pretraining	DeCoGLM, DeGLM, CoGLM
CLang8(EN)	2,372,119	Fine-tuning	DeCoGLM, DeGLM, CoGLM
FCE *all*	33,236	Fine-tuning	CoGLM (10B)
NUCLE	57,157	Fine-tuning	CoGLM (10B)
W&I+LOCNESS	34,308	Fine-tuning	CoGLM (10B)
Lang8 (CH)	1,092,285	Fine-tuning	All
HSK	95,320	Fine-tuning	All
FCGEC *train*	36,341	Fine-tuning	CoGLM (10B)
BEA19 *dev*	4,384	Validation	-
MuCGEC *dev*	1,137	Validation	-
FCGEC *dev*	2,000	Validation	-
CoNLL-14 *test*	1,312	Testing	-
BEA19 *test*	4,477	Testing	-
MuCGEC *test*	6,000	Testing	-
FCGEC *test*	3,000	Testing	-

Table 5: Dataset statistics. The rightmost column indicates the models that utilize the respective dataset; "All" signifies that DeCoGLM, DeGLM, CoGLM, and CoGLM (10B) all used the dataset as the training set.

In the experiments described in Section 4.1, the datasets used are outlined in Table 5. Due to constraints on our computational resources, the CoGLM (10B) models are fine-tuned on relatively smaller datasets, and the models are not pre-trained on synthetic datasets.

A.2 Dataset Examples

In Sections 3.1 and 3.2, we describe the construction of training data. By aligning the source text with the target text, we derive error detection labels and masked text, thereby constructing training samples as illustrated in Figure 2. In Section 3.3, we elaborate on a two-stage supervised fine-tuning approach, where the training data for the second stage is reconstructed based on the detection predictions made by the model trained in the first stage. During data construction, model-induced false positives for ERROR and INSERT are incorporated to generate new masked text and corresponding text pieces. It is crucial to note that this process is solely aimed at creating new masked text to enhance the model's ability to address false positives during the correction phase, while the detection labels used in training remain unchanged. Examples of the constructed training data are provided in Table 6, where "<s>" denotes the "begin of sentence" token and "</s>" represents the "end of sentence" token. For the sake of brevity, these tokens are omitted in the content of this paper except in Figure 2.

B Details of Experiments

B.1 Loss Weight

We pre-determine the weights in multi-task learning by intuitively observing the scales of two losses. This preliminary experiment was conducted on the CLang8 dataset, and the loss curves are depicted in Figure 5. It is evident from the figure that the detection loss ℓ_D and correction loss ℓ_C differ by roughly an order of magnitude. Consequently, we initially set $w_D = 10$, determine the weights for ERROR and INSERT categories in Focal Loss denoted by α_{EI}, and subsequently test whether $w_D = 10$ is an optimal choice, as discussed in Section 5.1.

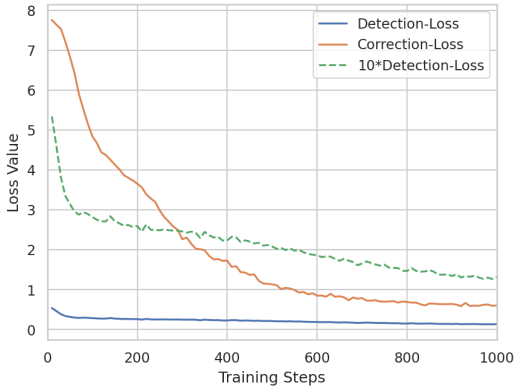

Figure 5: Loss curves in standard training condition.

Stage	Items	Example 1	Example 2
SFT1	Source Text $\boldsymbol{x}_s$	<s>The every male employees were standing in the back row .</s>	<s>They are covered with rust so bad .</s>
	Target Text $\boldsymbol{y}$	<s>All the male employees were standing in the back row .</s>	<s>They are covered with rust so badly .</s>
	Masked Text $\boldsymbol{x}_m$	<s>[MASK] male employees were standing in the back row .</s>	<s>They are covered with rust so [MASK] .</s>
	Text Pieces Input	<|startofpiecel> All the	<|startofpiecel> badly
	Text Pieces Target	All the <|endofpiecel>	badly <|endofpiecel>
	Detection Labels	K E E K K K K K K K K K	K K K K K K K E K K
SFT2	Detections by SFT1	K E E K E E K K K K K K	K K K K K I K K K K
	Merged Detecions	K E E K E E K K K K K K	K K K K K I K E K K
	Masked Text $\boldsymbol{x}'_m$	<s>[MASK] male [MASK] standing in the back row.</s>	<s>They are covered with rust [MASK] so [MASK] .</s>
	Text Pieces Input	<|startofpiecel> All the <|startofpiecel> employees were	<|startofpiecel> <|startofpiecel> badly
	Text Pieces Target	All the <|endofpiecel> employees were <|endofpiecel>	<|endofpiecel> badly <|endofpiecel>

Table 6: Examples of training data from CLang8 dataset in two fine-tuning stages. In detection labels, K=KEEP, E=ERROR and I=INSERT.

Configuration	EN Pretrain	EN finetune	CH Pretrain	CH finetune
DeCoGLM-Training				
Backbone	GLM-RoBERTa-large (Du et al., 2022)		GLM-large-chinese (Du et al., 2022)	
Backbone Parameters	335M		335M	
Batch size	12	12	12	12
Update frequecy	10	20	8	8(M), 10(F)
Max epochs	(20M iterations)	20	2	10(M), 20(F)
Evaluation key (SFT1)	-	AD-Accuracy	AD-Accuracy	AD-Accuracy
Evaluation key (SFT2)	-	General-Accuracy	-	General-Accuracy
Evaluation interval	10000	2000	4000	2000(M), 200(F)
Early stop	-	10	-	10
Max source text length	128	128	128	128
Warm-up steps (SFT1)	10000	1000	1000	1000(M), 200(F)
Warm-up steps (SFT2)	-	1000	-	1000(M), 200(F)
Weight Decay	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Learning rate scheduler	Polynomial	Polynomial	Polynomial	Polynomial
Learning rate (SFT1)	2×10^{-5}	3×10^{-6}	2×10^{-5}	1×10^{-5} (M), 4×10^{-5}(F)
Learning rate (SFT2)	-	1×10^{-6}	-	5×10^{-6} (M), 1×10^{-5}(F)
DeCoGLM-Inference				
KEEP threshold	0.38		None	
ERROR lower bound	0.5		None	
INSERT lower bound	0.6		None	
Beam size	3		3	
Max tokens per piece	10		10	

Table 7: The model hyper-parameters of proposed DeCoGLM. Both pretraining and fine-tuning configurations are presented. EN and CH represent English models and Chinese models, respectively. In the settings of Chinese fine-tuned models, M and F represent models for MuCGEC and FCGEC, respectively. The bottom of the table presents the hyper-parameters of inference.

B.2 Model Configurations

The training configurations for the integrated detection-correction model (DeCoGLM) and the parameters used during inference are presented in Table 7. To conserve computational resources during training, early stopping is employed, which requires the pre-definition of evaluation metrics on the validation set. Two primary metrics are utilized: (1) AD-Accuracy, defined as the sum of the recall for ERROR and INSERT and the accuracy of next token prediction by GLM, aiming to reinforce the aggressive detection principle mentioned in Section 3.1; (2) General-Accuracy, the geometric mean between the recall for the three detection labels and the accuracy of next token prediction by GLM. The configurations for training the separate models, DeGLM and CoGLM, are similar to those in Table 7. The pre-trained models include `glm-roberta-large`, `glm-large-chinese`, `glm-10b`, and `glm-10b-chinese`, accessible through HuggingFace[2]. We implement all the designed models using PyTorch, including DeCoGLM, DeGLM, and CoGLM.

All models are trained by the Trainer from the transformers[3] package in Python, on NVIDIA RTX 4090 GPUs. Due to resource constraints, all experiments are conducted with a fixed random seed (111), and single-run results are reported. We adopt the approach recommended by Rothe et al. (2021)

[2]https://huggingface.co

[3]https://huggingface.co/docs/transformers/index

Mode	Prompt
ZeroShot	Reply with a corrected version of the input sentence with all grammatical and spelling errors fixed. If there are no errors, reply with a copy of the original sentence. Input sentence: [TEXT] Corrected sentence:
+DeGLM	Reply with a corrected version of the input sentence with all grammatical and spelling errors fixed. If there are no errors, reply with a copy of the original sentence. Hint: We have detected some possible grammatical errors and replaced every error span with a [MASK] to get a masked sentence, you can reference the masked sentence to give final corrected sentence. If there is no [MASK] in the masked sentence, it means that we have not detected any grammatical errors in the input sentence. Input sentence: [TEXT] Masked Sentence: [MASKED_TEXT] Corrected sentence:

Table 8: GPT-4 prompts used in experiments, following Coyne et al. (2023).

		$F_{0.5}$ on test set		Average inference time per sample (ms)		
Backbone	**Structure**	**CoNLL-14**	**BEA-19**	**Detection**	**Correction**	**Total**
GLM-Roberta	De-Co	**64.71**	**70.85**	**14.5**	69.1	83.6
BART-large	De-Co	62.24	67.97	17.1	**43.4**	**60.5**
BART-large	Seq2Seq	64.46	67.94	-	266.2	266.2

Table 9: Time consumed in inference. De-Co represents the proposed detection-correction structure.

to post-process the model's predictions on English test datasets, aiming to ensure greater alignment of tokenization with the evaluation data.

B.3 GPT-4 Prompts

The prompts utilized during the inference of GPT-4 are illustrated in Table 8. For the Chinese tasks, the prompts are the direct translation of the corresponding English prompts. The API version of GPT-4 used in this paper is Preview-0315.

C Inference Speed

We conduct a brief evaluation of the inference speed of our proposed detection-correction structure, and the average inference speeds on the CoNLL-14 and BEA-19 test sets are presented in Table 9. The models are trained exclusively on the CLang8 dataset, and during the inference phase, no hyperparameters are adjusted, utilizing only beam search. Our proposed model achieves slightly better performance while maintaining a faster inference speed ($\approx$3x) than the Seq2Seq model. The experiments are conducted on an NVIDIA RTX 4090 GPU, with the same constrained batch size of 1 during inference.