Improving Grammatical Error Correction via Contextual Data Augmentation

Anonymous ACL submission

Abstract

Nowadays, data augmentation through synthetic data has been widely used in the field of Grammatical Error Correction (GEC) to alleviate the problem of data scarcity. However, these synthetic data are mainly used in the pre-training phase rather than the data-limited fine-tuning phase due to inconsistent error distribution and noisy labels. In this paper, we propose a synthetic data construction method based on contextual augmentation, which can ensure an efficient augmentation of the original data with a more consistent error distribution. Specifically, we combine rule-based substitution with model-based generation, using the generation model to generate a richer context for the extracted error patterns. Besides, we also propose a relabeling-based data cleaning method to mitigate the effects of noisy labels in synthetic data. Experiments on CoNLL14 and BEA19-Test show that our proposed augmentation method consistently and substantially outperforms strong baselines and achieves the state-of-the-art level with only a few synthetic data.

1 Introduction

Grammatical Error Correction (GEC) aims to detect and correct grammatical errors in a text (Wang et al., 2020; Bryant et al., 2022). It is a challenging task with a wide range of application scenarios, including search engines, writing assistants (Omelianchuk et al., 2020), and Automatic Speech Recognition (ASR) systems. Due to the low frequency of grammatical errors in real corpus, obtaining and annotating a certain number of high-quality GEC datasets is usually difficult and costly. Therefore, the currently available high-quality annotated GEC data is very limited (Ye et al., 2023), making synthetic data an important research direction for the data-starved task.

Nowadays, using synthetic data or data augmentation to improve the performance of GEC models has become a mainstream approach (Madnani et al., 2012; Grundkiewicz and Junczys-Dowmunt, 2014; Grundkiewicz et al., 2019). Common construction methods can be categorized into rule-based substitution (Awasthi et al., 2019; Choe et al., 2019) and model-based generation methods (Xie et al., 2018; Lichtarge et al., 2019; Zhou et al., 2019). However, the synthetic data constructed by the above methods are mainly used in the pre-training phase to initialize a better GEC model. The data augmentation methods used for the data-limited fine-tuning phase are of great research value.

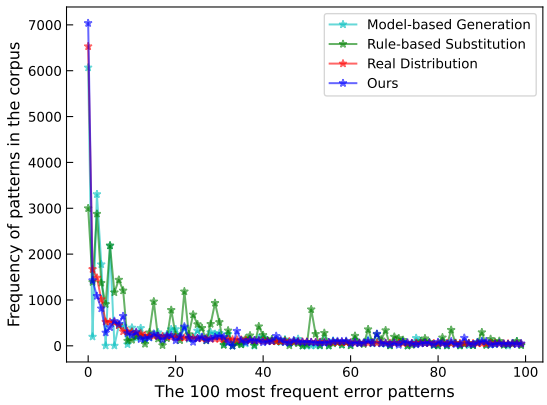

Figure 1: Illustration of the distribution of error patterns in each dataset. The x-axis represents the 100 most frequent error patterns in the annotated dataset W&I+L, and the y-axis represents the frequency of that error in the corresponding synthetic dataset.

There are two main reasons why previous synthetic data cannot apply to joint training in the fine-tuning phase. (1) **Inconsistent Error Distribution**. The high randomness of synthetic data makes it difficult to perfectly match the distribution of a certain high-quality data, leading joint training to performance degradation. Rule-based substitution methods are limited by the distribution and word frequency of the unlabeled corpus. Although

Wrong Sentence		Public transport enables our body to move one place to another.
Correct Sentence		Public transport enables our body to move from one place to another.
1-gram Aug	Pattern	$\emptyset \rightarrow$ from
	Source	They are coming $\emptyset$ the city center.
	Target	They are coming from the city center.
3-gram Aug	Pattern	move one $\rightarrow$ move from one
	Source	They move one place to another.
	Target	They move from one place to another.
5-gram Aug	Pattern	to move one place $\rightarrow$ to move from one place
	Source	They will have to move one place to another in order to find the treasure.
	Target	They will have to move from one place to another in order to find the treasure.

Table 1: An example of the proposed contextual augmentation approach. It achieves the effect of data augmentation by using a model to re-generate context for error patterns extracted from an existing parallel corpus. Wrong sentence and correct sentence are taken from the existing dataset. We extract error pattern of varying lengths from it. N-gram Aug represents the results of augmentation for error pattern of different lengths, where source represents the ungrammatical sentence, target represents the corretion, red represents grammatical errors and green represents correction results.

model-based generation methods can generate different types of grammatical errors (Stahlberg and Kumar, 2021), they still do not have stable controllability for specific errors with a small amount of synthetic data. As shown in Figure 1, the distribution of our proposed augmentation method is most consistent with the original dataset. (2) **Noisy Label**. Synthetic data is not human-labeled and cannot avoid introducing some mislabeling (inappropriate substitution or ungrammatical generation). For example, "I think you are right" may be incorrectly annotated as "I think that you are right" in synthetic data. As a text generation task with token-level metrics, the GEC task is very sensitive to this type of noise. Directly joint training of synthetic and real data brings serious performance degradation (Zhang et al., 2019). Recently, Ye et al. (2023) propose the MixEdit framework for grammatical error augmentation of the fine-tuning stage through pattern replacement. But it is still suffering from the two problems mentioned above.

In this paper, we propose a high-quality synthetic data construction method for the fine-tuning phase based on contextual augmentation. It can be viewed as a combination of a rule-based substitution approach and a model-based generation approach, where the model is utilized to generate a rich context for the extracted error patterns. An example of the augmentation data is shown in Table 1. Specifically, we first extract the error patterns (containing correct and incorrect token pairs) present in the real corpus through a GEC tool (ERRANT) and construct a corresponding error pattern pool. After that, we sample error patterns from the pool based on the true frequency of the original dataset to regenerate contexts for them. We regard this process as a hard constraint generation task, allowing the model to generate contextual sentences containing the correct pattern, and then obtaining the wrong sentence by rule substitution. We attempt both GPT2 (Radford et al., 2019) supervised generation and LLaMA2-7b-chat (Touvron et al., 2023) few-shot generation for our experiments. Finally, we use the baseline GEC model to relabel the synthetic data for joint training to mitigate the noise in the synthetic data.

The main contributions of this paper can be summarized as follows:

- We propose a synthetic data construction method based on contextual augmentation, which can stably generate a rich context for specific grammatical errors.
- To mitigate the effect of noisy labels, we introduce the re-labeling method into the synthetic data which improves the performance of the GEC model in joint training.
- Experiments show that our approach effectively enhances the robustness and performance of the GEC model by augmenting the high-quality annotated data in the fine-tuning phase.

2 Method

The main flow of our proposed synthetic data construction method based on context augmentation is

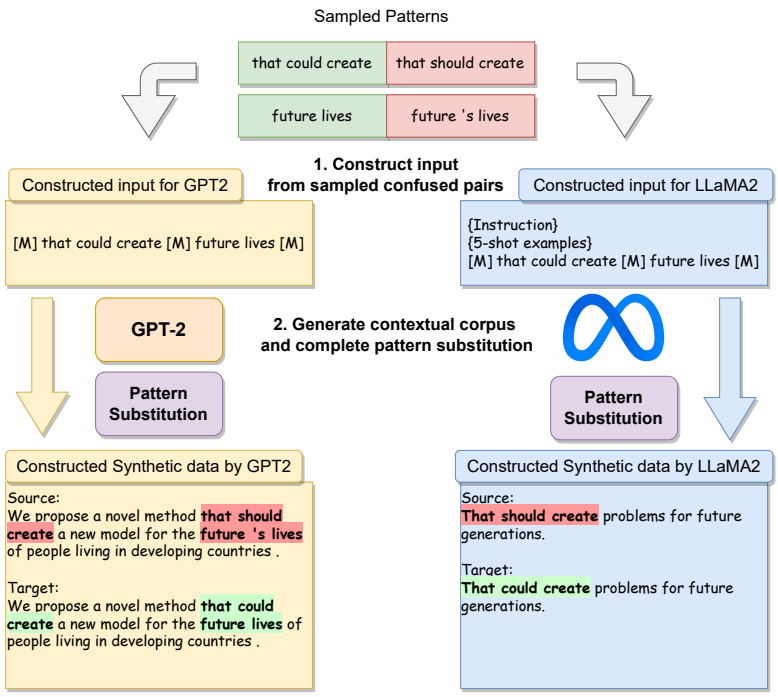

Figure 2: Illustration of synthetic data construction based on contextual augmentation. We uses both fine-tuned GPT2 and ICL of llama2 for the experiments. The red in the sampling patterns represents the wrong pattern and the green represents the correct pattern. Note that we combine the sampled **correct** patterns into a certain format for context generation, followed by pattern substitution to obtain a parallel corpus. Due to the sample decoding strategy, there may be cases where the context does not fully cover the pattern in the input as in the case of LLaMA. In practice, we generate parallel corpus by directly ignoring the unmatched patterns.

illustrated in Figure 2. First, we generate the synthetic data with contextual augmentation according to Section 2.1's method. After denoising by relabeling (Section 2.2), we use the synthetic data to augment the original data in the joint training (Section 2.3).

2.1 Pattern-based Context Generation

Both rule-based substitution and model-based generation methods generate synthetic data that require large amounts of data (in the millions) to guarantee a wide range of errors (Kiyono et al., 2019). However, in the supervised fine-tuning phase, the amount of data is very limited (W&I+L only includes about 30k), and millions of synthetic data for joint training is unrealistic. A more stable augmentation approach is needed to ensure that the original high-quality errors are adequately trained.

The main motivation for our proposed context augmentation is to leverage the modeling capability of language models to generate rich contexts for specific high-quality grammatical errors. Compared with other synthesis methods, we ensure that the error distribution is consistent with the original dataset through rule-based error patterns and the diversity of sample contexts through model-based generation. The synthesis method can be divided into three steps: building the error pattern pool (Section 2.1.1), synthesizing the contextual corpus (Section 2.1.2), and substituting 2.1.3 to get the parallel corpus.

2.1.1 Error Patterns Extraction

We follow Choe et al.'s (2019) setting and use the parsing tool ERRANT[1] to extract the editing operations present in the parallel corpus as error patterns according to the rules. We also extracted error patterns of different lengths for our experiments to ensure that the synthetic data contains more realistic errors. Excessively long patterns will make it difficult to match them in unlabeled text with the original rule-based substitution method, but the contextual augmentation-based generation method can solve this problem well.

We finally extract the patterns for each human-labeled GEC dataset, and merge the corresponding patterns into an error pattern pool for the construction of synthetic data. The statistics of extracted patterns can be found in Appendix A.

[1]https://github.com/chrisjbryant/errant

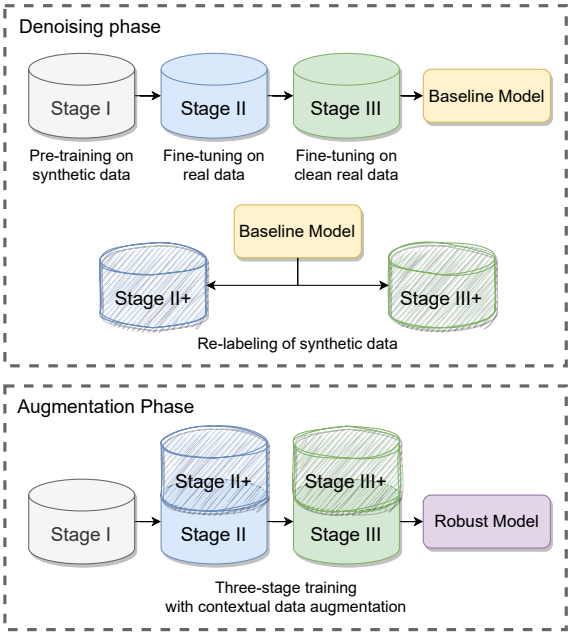

Figure 3: Illustration of the three phases of joint training with augmented data. We first denoise the synthetic data (Stage II+ & III+) using a baseline model trained in three stages, and subsequently conduct joint training to obtain a robust model.

2.1.2 Contextual Corpus Generation

With the error pattern pool and the frequency of the corresponding errors, we can simply obtain a set of pattern datasets with the same distribution as the annotated corpus by sampling. The goal of the generator model can be viewed as a hard-constrained text generation task (Welleck et al., 2019), generating a context that fully contains the target pattern. Considering that existing pre-trained models are trained on grammatically correct corpora, we only generate the corresponding contextual corpus based on the correct patterns and subsequently construct the parallel corpus by rule-based substitution.

In particular, the input to the model will be a combination of several randomly sampled patterns, which can be formulated as:

$$Pattern_{input} = Pattern_1\ [M]\ Pattern_2 \quad (1)$$

where $Pattern_i$ represents the correct pattern that was sampled and $[M]$ represents the context placeholder that needs to be generated. It should be noted that the number of patterns for each sample will be randomly selected between 1 and 2, and Equation 1 represents the case of 2 patterns only. The output of the model is a piece of text containing the corresponding input pattern.

In this paper, we experiment with two models as context generators, GPT2 and LLaMA2, representing the two settings of supervised fine-tuning and few-shot generation, respectively.

Finetuning for GPT2 We choose GPT2 as the backbone network to represent the performance of the fine-tuned generative model in the contextual augmentation task. The model generates the target corpus directly from the provided pattern, which can be formulated as:

$$S = Pattern_{input} \text{ <sep> } Sentence_{target} \quad (2)$$

where $Pattern_{input}$ is the combination of the patterns mentioned in Equation 1 and $Sentence_{target}$ is the target corpus containing all the patterns. <sep> is the special token dividing the input S into two parts.

For the training phase, we use the autoregressive way consistent with the pre-training:

$$\mathcal{L} = \sum_{k=i}^{j} -log(P(t_k|t_0t_1...t_{k-1};\theta)) \quad (3)$$

where θ is the set of parameters of the language model, i and j represent the start and the end index of $Sentence_{target}$, and t_i represents the i-th token in the model input S like Equation 2.

As for the training data, in order to ensure that the style of the generated text is consistent with the training data, we directly adopt the correct sentences in the non-native speaker GEC dataset C-Lang8 (Rothe et al., 2021) dataset as the target sentences to construct the training set. Specifically, we randomly replace multiple consecutive text segments in the corpus with $[M]$ label and train the model to generate the corresponding context based on the remaining text segments (error patterns during inference).

Few-shot generating for LLaMA2 Recently, LLMs (Brown et al., 2020; Wei et al., 2021; Touvron et al., 2023) have presented powerful in-context learning capabilities to accomplish complex NLP tasks based on a few example samples. Therefore we also try to use the LLM to generate a more appropriate context for the error patterns. We directly use the prompt with the 5-shot setting and ask the LLM to generate the conditional corpus. Details of the prompts and input format can be found in Appendix C. In addition, the GEC corpus is usually large, and considering the cost issue, we choose the open-source LLM LLaMA2-7b-chat (Touvron et al., 2023) for our experiments.

2.1.3 Pattern Substitution

Given the contextual corpus and error pattern pairs, we can obtain the corresponding GEC parallel corpus by simple substitution (Choe et al., 2019). To ensure the diversity of the corpus, we choose the sampling decoding strategy during generation, and we directly ignore the patterns that can't be matched exactly. Besides, we only substitute the error patterns for 50% of the synthetic data to ensure a consistent error rate with the annotated datasets during the joint training process.

2.2 Synthetic Data Denoising

Compared to human-annotated data, synthetic data is more accessible but inevitably noisy. Previous work (Zhang et al., 2019) has proven that direct joint training of synthetic and real data affects the metrics of the final model. Improper substitutions (grammatically correct both before and after substitution) are the main cause of noisy synthetic data. Yasunaga et al. (2021); Cao et al. (2023) propose some sentence-level filtering methods based on scores such as PPL, but the filtering granularity and accuracy are not sufficient in the joint training setting. We need an efficient way of filtering at the token level.

Inspired by Rothe et al.'s (2021) distillation method, which mitigates the effects of noisy data by relabeling the corpus with a powerful GEC model, we also want to denoise the corpus through relabeling. Specifically, we view the synthetic data as an unlabeled grammatical error-filled corpus and relabel it using a strong baseline model. Since the synthetic data is obtained by augmentation using the original dataset, relabeling using the original model effectively removes the noise while correcting most of the grammatical errors.

2.3 Joint Training Process

To obtain a strong baseline model, we follow Bout et al.'s (2023) approach of using three stages for training. Our proposed data augmentation method will also be applied to the stages of fine-tuning.

As shown in Figure 3, we divide the available dataset into three stages for training. We use $C4_{200M}$ (Stahlberg and Kumar, 2021) dataset for the pre-training phase (stage I), which generate grammatical errors with type distribution consistent with BEA-Dev (Bryant et al., 2019) based on the seq2edit model. For Stage II, we used the complete available annotated dataset (see Table 2 for details) to fine-tune the model, including Lang-8, NUCLE, FCE, and W&I+L. As the highest-quality annotated GEC dataset, we individually fine-tune the stage III on W&I+L.

Dateset	Errorful%	Sentences#	Usage
$C4_{200M}^{\star}$	99.4	∼180M	I
Lang-8	48.0	1,037,561	II
NUCLE	38.0	56,958	II
FCE	62.5	28,350	II
W&I+L	67.3	34,304	II&III
StageII-Syn*	50.0	2M	II+
StageIII-Syn*	50.0	200,000	III+
BEA19-Dev	64.3	4,384	Dev
Conll14-Test	71.9	1,312	Test
BEA19-Test	N/A	4,477	Test

Table 2: Statistical information on grammatical error correction datasets. Note that $\star$ indicates synthetic datasets. II+ and III+ represent the augmented dataset of corresponding stages, which will be mixed with real data for joint training in the proposed method.

Due to data distribution and quality, previously synthetic data are mainly utilized in the pre-training phase (stage I). In contrast, our proposed context augmentation approach is mainly used to adapt a small amount of high-quality fine-tuned data (stage II&III). We generate different amounts of synthetic data (StageII-syn and StageIII-syn in Table 2) for different stages using the corresponding error pattern pool. After that, we directly train the synthesized data jointly with the real data of fine-tuning stages, as shown in Figure 3.

3 Experiment

3.1 Setting

Datasets In Table 2, we summarize the statistical information for the all relevant datasets. The $C4_{200M}$ (Stahlberg and Kumar, 2021) dataset used for stage I is synthetic data based on Seq2Edit (Stahlberg and Kumar, 2020) model generation. For the other stages, we use the following common GEC datasets: Lang-8 Corpus of Learner English (Lang-8) (Mizumoto et al., 2011; Tajiri et al., 2012) collects from non-native speaker online learning websites Lang-8[2]; National University of Singapore Corpus of Learner English (NUCLE) (Dahlmeier et al., 2013) consists of essays written by undergraduate students on a variety of topics and annotated by professional English teachers; First Certificate in English (FCE) (Yannakoudakis

[2]https://lang-8.com/

Model	Model Size	CoNLL2014			BEA19-Test		
		Prec	*Rec*	$F_{0.5}$	*Prec*	*Rec*	$F_{0.5}$
GECToR$^{\diamondsuit}$ (Omelianchuk et al., 2020)	350M	77.5	40.1	65.3	79.2	53.9	72.4
T5-large$^{\heartsuit}$ (Rothe et al., 2021)	770M	-	-	66.1	-	-	72.1
T5-XL$^{\heartsuit}$ (Rothe et al., 2021)	3B	-	-	67.8	-	-	73.9
T5-XXL$^{\heartsuit}$ (Rothe et al., 2021)	11B	-	-	**68.9**	-	-	**75.9**
ShallowAD$^{\clubsuit}$ (Sun et al., 2021)	~240M	71.0	52.8	66.4	-	-	72.9
SynGEC$^{\heartsuit}$ (Zhang et al., 2022)	400M	74.7	49.0	67.6	75.1	65.5	72.9
TemplateGEC$^{\heartsuit}$ (Li et al., 2023)	770M	74.8	50.0	68.1	76.8	64.8	74.1
MixEdit$^{\heartsuit}$ (Ye et al., 2023)	400M	75.6	46.8	67.3	76.4	62.7	73.2
MultiTaskBART$^{\spadesuit}$ (Bout et al., 2023)	400M	75.4	51.2	**68.9**	78.2	65.5	75.3
BART Baseline$^{\spadesuit}$	400M	73.8	53.5	68.6	74.5	68.9	73.5
+ CDA w/o denoising	400M	76.7	44.8	67.1	76.1	67.3	74.1
+ CDA w/ denoising	400M	76.2	52.2	**69.8**	77.7	67.5	**75.4**

Table 3: The results of the strong BART Baseline initialized on $C4_{200M}$ and Context Date Augmentation (CDA) methods for the single model. CDA w/o denising means directly using the raw synthetic data constructed by the proposed method without relabeling. In addition to publicly available annotated datasets, existing GEC models also use: $\diamondsuit$ rule-based synthetic data from one-billion-word (9M), $\heartsuit$ cleaned version of Lang8 (2.4M), $\clubsuit$ model-based synthetic data (300M), $\spadesuit$ model-based synthetic data ($C4_{200M}$).

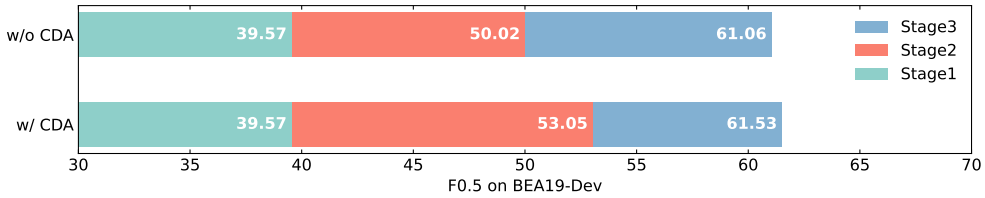

Figure 4: The results of the three-stage model on BEA19-Dev after contextual augmentation respectively.

et al., 2011) primarily contains answers to the upper-intermediate level exams written by English language learners; Write & Improve + LOCNESS Corpus (W&I+L) (Bryant et al., 2019) includes two parts of data. The Write & Improve dataset consists of chunks of text (articles, letters, etc.) submitted to the W&I system written by English learner; In contrast, LOCNESS consists of essays written by native English-speaking students and is used for evaluation purposes only.

In addition to the existing publicly available datasets, we constructed synthetic data for different stages (corresponding to StageX-Syn in the Table 2) by contextual augmentation using the proposed method. As for the amount of synthetic data, we heuristically choose 2M pairs for Stage II, 200,000 pairs for Stage III. We perform ablation experiments on the amount of augmented data in subsequent analyses.

Evaluation We use BEA19-Dev (Bryant et al., 2019) as a validation set to evaluate the performance of the GEC model. In the main experiments, we report results of Conll14-Test (Ng et al., 2014) using the official M2 scorer (Dahlmeier and Ng, 2012), and results of BEA19-Test (Bryant et al., 2019) using ERRANT (Bryant et al., 2017) on the online platform[3].

Training Details As described in Section 2.1.2, we use GPT2-base and LLaMA2-7b-chat as context generators for our experiments. For the implementation of the baseline GEC model, we refer to previous setups (Zhang et al., 2022) and use Seq2Seq-based BART-large (Lewis et al., 2019) for the experiments, which is trained using the Fairseq[4] framework. More details on hyperparameters can be found in Appendix B.

3.2 Baseline Approaches

We select several recent state-of-the-art methods as baselines for comparison. GECToR (Omelianchuk et al., 2020) is an efficient auto-encoder grammatical error correction model, which corrects errors

[3]https://codalab.lisn.upsaclay.fr/competitions/4057
[4]https://github.com/facebookresearch/fairseq

by predicting edit tags. Rothe et al. (2021) verify the performance of T5 (Raffel et al., 2020) models of various scales (from small to xxl) on the GEC task. SynGEC (Zhang et al., 2022) incorporates the syntactic information of the text into the model using GCN. TemplateGEC (Li et al., 2023) fuses the seq2edit and seq2seq models to provide a new two-stage framework for error detection and correction. Ye et al. (2023) propose a data augmentation approach MixEdit that strategically and dynamically augments realistic data, without requiring extra monolingual corpora. Bout et al. (2023) propose a multi-task pre-training method and optimization strategy, which greatly improved the performance of the GEC model.

In this paper, we use the three-stage training model initialized by the $C4_{200M}$ datasets as strong baselines. With Contextual Data Augmentation (CDA), we integrate contextually augmented synthetic data for training in the fine-tuning phase, as shown in Figure 3.

3.3 Main Experimental Results

The experimental results of our proposed augmentation method on CoNLL14 and BEA19 are shown in Table 3. We obtain a strong baseline model by training in three stages according to the Bout et al.'s (2023) setting. The results show that contextual data augmentation can effectively improve the robustness and generalization of the original model, and bring significant improvements on both CoNLL14 and BEA19-Test datasets. Our 400M BART model achieves the state-of-the-art through contextual data augmentation, and is comparable to the 11B T5-XXL. In addition to this, we find that the impact of the augmented data's noisy labels can be well mitigated by simple relabeling. It should be noted that the proposed method improves the modeling precision with a slight loss in recall, which is encouraged in GEC tasks since ignoring an error is not as bad as proposing a wrong correction (Ng et al., 2014).

4 Analysis

4.1 Impact of Different Generators

In this article, we have experiment with two generator settings, GPT2 fine-tuning, and LLaMA2 ICL, to generate synthetic data. The GPT2 fine-tuning model is relatively small, which has a faster generation efficiency and follows the task requirements better after training. On the contrary, LLaMA2 has a larger number of parameters and generates more diverse and fluent texts. But the model generates more slowly and follows the instructions more weakly.

Method	BEA19-Dev		
	Prec	*Rec*	$F_{0.5}$
Stage2 Model	61.28	28.84	50.02
+ Stage3 GPT2	64.16	**51.13**	61.05
+ Stage3 LLaMA2	**64.23**	51.07	**61.08**
Stage2 Model	61.28	28.84	50.02
+ Stage3 1-gram	63.73	**52.50**	61.12
+ Stage3 3-gram	**64.39**	51.07	**61.20**
+ Stage3 5-gram	64.16	51.13	61.05

Table 4: Ablation study of the different generators and the different pattern lengths on BEA19-Dev.

To verify the effect of the different generators on the quality of the generated text, we use them to generate 200k synthetic data on the high-quality text of stage III respectively for joint training. The experiment results are shown in Table 4. We find similar conclusions to Xu et al. (2023), that whether the synthetic data generated by the fine-tuned model or the LLM in-context learning has little effect on the final model performance. So we mainly use GPT2 fine-tuning as the generator for experiments for efficiency considerations.

4.2 Impact of Pattern Length

Unlike previous rule-based substitution approaches (Choe et al., 2019), our proposed method is not restricted to the distribution of the unlabeled corpus, so the length of the error pattern is no longer limited to the token level. To obtain the optimal pattern length, we experiment with contextual augmentation using 1-gram, 3-gram, and 5-gram patterns as shown in Table 1. The results are shown in Table 4, the model has the best performance with synthetic data generated by the 3-gram pattern. We hypothesize that appropriately long patterns make the synthetic data distribution closer to the source data, indirectly improving the quality of the relabeling.

4.3 Impact of Data Mixing Ratio

In joint training, the impact of the ratio of synthetic to real data is significant. We conduct joint training experiments with different amounts of synthetic data, and the results are shown in Figure 5. Consistent with our analysis, the ratio of synthetic data to real data is a trade-off. As the amount of synthetic

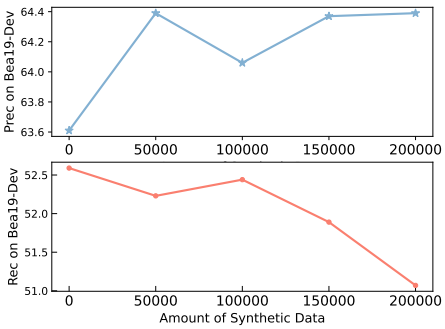

Figure 5: The effect of different amounts of synthetic data in joint training on the final system.

Synthetic Data	BEA19-Dev		
	Prec	*Rec*	$F_{0.5}$
N/A	63.61	52.59	61.06
PIE	63.89	51.68	61.01
$C4_{200M}$	63.96	51.42	60.98
CDA (ours)	**64.39**	**52.23**	**61.53**

Table 5: Performance of different synthetic data participating in joint training after denoising. N/A means the no data augmentation situation.

data increases, the precision of the system gradually increases. At the same time, the proportion of higher-quality real data has declined, possibly leading to some decline in error recall.

4.4 Quality Assessment of Synthetic Data

To compare the quality of our proposed context augmentation with other synthetic data. We adopt the same three-stage training and denoising for synthetic data of different construction methods. We choose PIE (Awasthi et al., 2019) and $C4_{200M}$ (Stahlberg and Kumar, 2021) as our baselines, which represent the rule-based substitution method and the model-based generation method, respectively. The results are shown in Table 5, where our proposed CDA method is able to better fit the distribution of the original dataset within the limited data, thus improving the model performance in joint training.

5 Related Work

5.1 Synthetic Data for GEC Task

Construction of Synthetic Data As a data-starved field, synthetic data has been proven effective in improving the GEC systems (Kiyono et al., 2019), which can be categorized into rule-based substitution and model-based generation methods. Rule-based methods are mainly constructed through direct noise addition (Xu et al., 2019; Zhou et al., 2019; Kiyono et al., 2020), pattern substitution (Choe et al., 2019), and parsing tools (Grundkiewicz et al., 2019). Model-based generation methods mainly include back-translation (Xie et al., 2018; Stahlberg and Kumar, 2021) and round-trip translation (Zhou et al., 2019) based on Seq2Seq architecture. (Stahlberg and Kumar, 2021) propose a synthesis method based on the Seq2Edit (Stahlberg and Kumar, 2020) architecture capable of generating synthetic data by specifying grammatical error types.

Utilization of Synthetic Data Zhang et al. (2019) have explored the use of synthetic data through detailed experiments. The experiments prove that using synthetic data in the pre-training phase achieves optimal results. Some unsupervised grammatical error correction work (Yasunaga et al., 2021; Cao et al., 2023) have used the self-training framework to label and co-train unlabeled error corpus to obtain an improvement in effectiveness.

5.2 LLM for GEC Task

LLMs (Brown et al., 2020; Wei et al., 2021; Touvron et al., 2023) have made significant improvements in a wide range of natural language processing tasks. However, LLMs do not perform well on common benchmarks (Coyne et al., 2023) due to traditional evaluation metrics and over-corrections. Although Fang et al. (2023) have demonstrated that some improvement is achieved by few-shot and chain-of-thought settings, there is still a big gap between LLMs and traditional fine-tuning models. In view of this, using LLM to construct synthetic data for the GEC task (Fan et al., 2023) can be considered as another feasible direction.

6 Conclusion

In this paper, we propose a synthetic data construction method based on contextual augmentation. It stably augments the context of the source data and ensures a consistent error distribution. Previous methods suffer from noisy labels of synthetic data. We significantly improve the performance of synthetic data in joint training through a re-labeling-based denoising method. We validate the effectiveness of our proposed method on several common datasets.

Limitations

Firstly, compared to other current synthetic data construction methods, generating synthetic data based on contextual augmentation takes more time and resources. For each sample, we need to complete the inference process on both sides of context generation and denoising. Secondly, the re-predicted results are not completely correct and can only alleviate noise. There are still a small number of incorrect labels in the synthetic data. In addition, it should be noted that the context augmentation method we proposed can only provide richer context for errors existing in the annotated dataset, and cannot introduce new grammatical errors. We will focus on investigating how to better generate high-quality synthetic data that contains a wider variety of grammatical errors utilizing LLMs in our future work.

Ethics Statement

In this paper, we explore the application of contextual augmentation-based synthetic data on the GEC task. The source data for these methods come exclusively from publicly available project resources on legitimate websites and do not involve any sensitive information. In addition, all baselines and datasets used in our experiments are also publicly available, and we have acknowledged the corresponding authors by citing their work.

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.

Andrey Bout, Alexander Podolskiy, Sergey Nikolenko, and Irina Piontkovskaya. 2023. Efficient grammatical error correction via multi-task training and optimized training schedule. *arXiv preprint arXiv:2311.11813*.

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.

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. *Computational Linguistics*, pages 1–59.

CJ Bryant, Mariano Felice, and Edward Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. Association for Computational Linguistics.

Hannan Cao, Liping Yuan, Yuchen Zhang, and Hwee Tou Ng. 2023. Unsupervised grammatical error correction rivaling supervised methods. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3072–3088.

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. *arXiv preprint arXiv:1907.01256*.

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.

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.

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

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

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2014. The wiked error corpus: A corpus of corrective wikipedia edits and its application to grammatical error correction. In *Advances in Natural Language Processing: 9th International Conference on NLP, PolTAL 2014, Warsaw, Poland, September 17-19, 2014. Proceedings 9*, pages 478–490. Springer.

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.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. *arXiv preprint arXiv:1909.00502*.

Shun Kiyono, Jun Suzuki, Tomoya Mizumoto, and Kentaro Inui. 2020. Massive exploration of pseudo data for grammatical error correction. *IEEE/ACM transactions on audio, speech, and language processing*, 28:2134–2145.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinghao Li, Xuebo Liu, Shuo Wang, Peiyuan Gong, Derek F Wong, Yang Gao, He-Yan Huang, and Min Zhang. 2023. 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.

Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. *arXiv preprint arXiv:1904.05780*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Exploring grammatical error correction with not-so-crummy machine translation. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 44–53.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 147–155.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

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

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

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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

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.

Felix Stahlberg and Shankar Kumar. 2020. Seq2edits: Sequence transduction using span-level edit operations. *arXiv preprint arXiv:2009.11136*.

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.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. *arXiv preprint arXiv:2106.04970*.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for esl learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 198–202.

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, Jie Liu, and Zhuo Liu. 2020. A comprehensive survey of grammar error correction. *arXiv preprint arXiv:2005.06600*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Sean Welleck, Kianté Brantley, Hal Daumé Iii, and Kyunghyun Cho. 2019. Non-monotonic sequential text generation. In *International Conference on Machine Learning*, pages 6716–6726. PMLR.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Y Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. 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 619–628.

Benfeng Xu, Quan Wang, Yajuan Lyu, Dai Dai, Yongdong Zhang, and Zhendong Mao. 2023. S2ynre: Two-stage self-training with synthetic data for low-resource relation extraction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8186–8207.

Shuyao Xu, Jiehao Zhang, Jin Chen, and Long Qin. 2019. Erroneous data generation for grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 149–158.

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.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2021. Lm-critic: Language models for unsupervised grammatical error correction. *arXiv preprint arXiv:2109.06822*.

Jingheng Ye, Yinghui Li, Yangning Li, and Hai-Tao Zheng. 2023. Mixedit: Revisiting data augmentation and beyond for grammatical error correction. *arXiv preprint arXiv:2310.11671*.

Yi Zhang, Tao Ge, Furu Wei, Ming Zhou, and Xu Sun. 2019. Sequence-to-sequence pre-training with data augmentation for sentence rewriting. *arXiv preprint arXiv:1909.06002*.

Yue Zhang, Bo Zhang, Zhenghua Li, Zuyi Bao, Chen Li, and Min Zhang. 2022. Syngec: Syntax-enhanced grammatical error correction with a tailored gec-oriented parser. *arXiv preprint arXiv:2210.12484*.

Wangchunshu Zhou, Tao Ge, Chang Mu, Ke Xu, Furu Wei, and Ming Zhou. 2019. Improving grammatical error correction with machine translation pairs. *arXiv preprint arXiv:1911.02825*.

A Error Pattern Information

We have extracted the error patterns from the existing annotated dataset using the ERRANT tool (Bryant et al., 2017), as described in Section 2.1.1. The number of patterns for each dataset is shown in Table 8. We maintain a separate error pattern pool for each dataset, from which we sample each time to generate synthetic data.

Configuration	Value
Stage1	
Backbone	BART-large (Lewis et al., 2019)
Devices	4 Tesla V100S-PCIE-32GB
Epochs	10
Max tokens	4096
Update freq	8
Optimizer	Adam (Kingma and Ba, 2014)
Learning rate	3e-05
Max source length	1024
Dropout-src	0.2
Clip norm	0.1
Label smoothing	0.1
Stage2	
Epochs	20
Learning rate	1e-05
Warmup-updates	2000
Patient	5
Stage3	
Epochs	50
Learning rate	3e-06
Warmup-updates	200
Patient	10

Table 6: Hyperparametric details of the BART-based three-stage GEC model. In Stage II,III only the parameters that differ from those in Stage1 are described.

B Hyper-parameters

We illustrate the hyper-parameters during training of the baseline model (see Table 6 for details) and the GPT2-based generative model (see Table 9 for details) here.

C Instruction Format for Synthetic Data Generation

When using LLaMA2-7b-chat for synthetic data generation, we use the 5-shot setting to generate the corresponding context for the sampled error patterns. We have given an example to illustrate the specific input format as shown in Table 7.

Instruction	[INST] < <SYS> > You are a helpful assistant.< </SYS> > Use phrases from input to make sentences. You should fill in [M] to make input sentence more complete. You can't change any form or order of the words in input. Make sure you fully use the phrases in #input. [/INST]
5-shot	#input: [M] sized city with eighty thousand [M] #output: My town is a medium - sized city with eighty thousand inhabitants . #input: [M] my own plan too , [M] to be the same as them . [M] #output: I have my own plan too , but I do n't want to be the same as them . I want to become a journalist . #input: Nowadays , each family has more than 1 [M] one of several reasons why [M] #output: Nowadays , each family has more than 1 car for each person , this is only one of several reasons why people use less public transport . #input: [M] they might want to safeguard [M] #output: On the other hand , they might want to safeguard the national image . #input: Lucy , Molly , and [M] a cowboy , and a [M] #output: Lucy , Molly , and their parents , a cowboy , and a teacher .
Input	#input: And I went [M] important [M]
Output	#output: And I went to the library to study for an important exam .

Table 7: An example input format for LLaMA2 ICL synthetic data generation

Dataset	Sentence#	Error Pattern#
Lang-8	1,037,561	677,475
NUCLE	56,958	49,347
FCE	28,350	43,854
W&I+L	34,304	62,952

Table 8: Statistics of the error pattern pool for each dataset.

Configuration	Value
Backbone	GPT2-base (Radford et al., 2019)
Devices	4 Tesla V100S-PCIE-32GB
Epochs	20
Batch size	32
Update freq	4
Optimizer	AdamW (Loshchilov and Hutter, 2017)
Learning rate	5e-05
Max length	256
Warmup ratio	0.1

Table 9: Hyperparametric details of the GPT2-based contextual generator.