FCGEC: Fine-Grained Corpus for Chinese Grammatical Error Correction

Lvxiaowei Xu[1], **Jianwang Wu**[1], **Jiawei Peng**[1], **Jiayu Fu**[2], **Ming Cai**[1*]
[1]Department of Computer Science and Technology, Zhejiang University
[2]Base Station Platform Software Development Dept, Huawei Co., Ltd.
[1]{xlxw, wujw, pengjw, cm}@zju.edu.cn
[2]fionafu0808@gmail.com

Abstract

Grammatical Error Correction (GEC) has been broadly applied in automatic correction and proofreading system recently. However, it is still immature in Chinese GEC due to limited high-quality data from native speakers in terms of category and scale. In this paper, we present FCGEC, a fine-grained corpus to detect, identify and correct the grammatical errors. FCGEC is a human-annotated corpus with multiple references, consisting of 41,340 sentences collected mainly from multi-choice questions in public school Chinese examinations. Furthermore, we propose a Switch-Tagger-Generator (STG) baseline model to correct the grammatical errors in low-resource settings. Compared to other GEC benchmark models, experimental results illustrate that STG outperforms them on our FCGEC. However, there exists a significant gap between benchmark models and humans that encourages future models to bridge it. Our annotation corpus and codes are available at https://github.com/xlxwalex/FCGEC[†].

1 Introduction

Grammatical error correction (GEC) is a complex task, aiming at detecting, identifying and correcting various grammatical errors in a given sentence. GEC has recently attracted more attention due to its ability to correct and proofread the text, which can serve a variety of industries such as education, media and publishing (Wang et al., 2021b).

However, Chinese GEC (CGEC) is still confronted with the following three obstacles: (1) **Lack of data.** The major obstacle in CGEC is that the high-quality manually annotated data is limited compared to other languages (Dahlmeier et al., 2013; Napoles et al., 2017; Rozovskaya and Roth, 2019; Bryant et al., 2019; Flachs et al., 2020; Trinh and Rozovskaya, 2021). There are only five publicly accessible datasets in CGEC: NLPCC18 (Zhao et al., 2018) , CGED (Rao et al., 2020), CTC-Qua, YACLC (Wang et al., 2021a) and MuCGEC (Zhang et al., 2022). (2) **Data sources are non-native speakers.** The sentences in NLPCC18, CGED, YACLC and MuCGEC are all collected from Chinese as a Foreign Language (CFL) learner sources. However, massive errors from native speakers rarely arise in these sources. Therefore, the native speaker errors are more challenging with the inclusion of pragmatic data. Though CTC-Qua covers grammatical errors in native speakers, it has insufficient scale with 972 sentences. (3) **Limited multiple references.** For an erroneous sentence, there tends to be different correction methods. The sentences revised by the model may be correct, but different from the ground truth. This may cause unexpected performance degradation (Bryant and Ng, 2015). Besides, more references can offer various correction schemas enabling the model to accommodate more scenarios. Among CGEC, only MuCGEC and YACLC provide rich references.

To tackle aforementioned obstacles, we present FCGEC, a large-scale fine-grained GEC corpus with multiple references. The sentences in FCGEC are mainly collected from multi-choice questions in public school Chinese examinations. Therefore, our FCGEC is more challenging since it involves more pragmatic data in the examinations of native speakers. As for multiple references, we assign 2 to 4 annotators on each sentence, thus more references can be attained in this way. Moreover, we generate more references in the annotation process through techniques with synonym substitution.

In order to correct the grammatical errors, recent works are mostly based on two categories of benchmark models. **Sequence-to-sequence (Seq2Seq)** approaches regard GEC as a generation task that straightforward converts an erroneous sentence to the correct one (Yuan and Briscoe, 2016; Zhao and Wang, 2020; Fu et al., 2018). However, training

* Corresponding author.
† Online evaluation site: https://codalab.lisn.upsaclay.fr/competitions/8020.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 1900–1918

such a generation model requires more computational resources due to the autoregressive decoder. Moreover, the generated style of Seq2Seq models is more arbitrary, which is not well applicable for GEC task. More recently, **sequence-to-edit (Seq2Edit)** approaches gain interests which take GEC as a token-level labeling task (Awasthi et al., 2019; Omelianchuk et al., 2020) via different edits, such as *insert*, *delete*, etc. Nevertheless, previous work falls short on altering the word order and correcting errors simultaneously with iterating.

To fill these gaps, we propose Switch-Tagger-Generator (STG) model as an effective baseline to correct grammatical errors in low-resource settings inspired by Mallinson et al. (2020). Our STG can be decomposed into three modules: *Switch* module determines the permutation of characters while *Tagger* module identifies the operation tags of each character in the sequence. Notably, benefiting from carefully designed compound tags, we eliminate the necessity for iteration. As for *Generator* module, we adopt non-autoregressive approach to fill in the characters that do not appear in the source.

In summary, our contributions are as follows:

1. We present FCGEC, a large-scale fine-grained corpus with multiple references and more challenging errors for CGEC.

2. We propose a STG model and then conduct experiments to compare with two categories of benchmark models (Seq2Seq and Seq2Edit). Experimental results illustrate that our STG model outperforms these models on FCGEC.

3. We find a significant gap between human performance and benchmark models that encourage future models to bridge it.

2 Corpus Construction

2.1 Data Collection

We collect raw sentences mainly from two resources to obtain various Chinese grammatical error corpus from native speakers. **(1) Public examination websites.** We crawl the multi-choice grammatical error problems (More erroneous sentences than correct sentences) through public websites which contain exercises and exams designed by teachers and experts. These problems cover public school examinations for native students from elementary to high school. **(2) News aggregator sites.** To balance the quantity of erroneous sentences and correct sentences, we attain a diverse range of high quality sentences without grammatical error in news aggregator sites.

In total, we collect 54,026 raw sentences from these resources. After removing duplicated or incomplete sentences, there are 41,340 sentences in our FCGEC corpus. We describe in detail the data sources and data structures in Appendix A & B.

2.2 Fine-grained Data Format

To facilitate model for grammatical error detection and correction, we designate three-tier hierarchical levels of golden labels in FCGEC as follows:

Detection Level. As a preliminary procedure to correcting grammatical errors, we require the binary classification of a given sentence according to whether it contains grammatical errors or not.

Identification Level. The labels in this level could be regarded as necessary for a multi-class categorization problem. As the examples shown in Table 1, we group grammatical errors into seven categories based on the error hierarchy. The definition of error types are as follows: **Incorrect Word Collocation (IWC)** is a word-level grammatical error in which the related words are combined in the improper pattern. **Component Missing (CM)** and **Component Redundancy (CR)** are also word-level errors that some components (e.g., subject and object) of the sentence are missing or redundant. **Structure Confusion (SC)** is a syntax-level grammatical error that combines two similar grammatical structures into a single incorrect one. **Incorrect Word Order (IWO)** covers grammatical errors in word-level and pragmatic-level. Compared to the previous work (Zhang et al., 2022), we also take into account the errors that require logic, common sense on top of syntax (e.g., recursive relationships). **Illogical (ILL)** and **Ambiguity (AM)** are pragmatic errors. The former comprises contradictory statements, while the latter includes expressions with indeterminate meanings.

Correction Level. In the correction level, we propose an *operation-oriented* paradigm to construct GEC labels instead of the *error-coded* or *rewriting* paradigms utilized in previous works (Ng et al., 2014; Sakaguchi et al., 2016). In *rewriting* paradigms, the annotators directly rewrite the raw sentences to the correct sentences without grammatical errors. However, it is difficult for annotators

Figure 1: An example of *operation-oriented* paradigm.

to rewrite in a consistent style, which leads to a drop in annotation quality. As for the *error-coded* paradigm, the annotators may diverge in determining the boundaries of the erroneous spans, thus raising the complexity of the procedure.

In contrast, the *operation-oriented* paradigm is on the basis of four fundamental correction operations : *Insert*, *Delete*, *Modify* and *Switch*. As an example shown in Figure 1, this paradigm is more compatible with the conventions of the annotator when correcting errors. Meanwhile, annotators only need to consider what operations are required to correct the sentences, instead of paying attention to the erroneous span (e.g., the selection of the words is left to post-processing for unified optimization). In addition, we have a large amount of correction prompts (explanations of grammatical error problems) developed by teachers and experts based on these four operations that can be utilized to accelerate annotation process.

2.3 Annotation Procedure

The annotators are asked to follow the given prompts to complete the three levels of labeling. Notably, we allow annotators to add unlimited references to sentences with grammatical errors based on the four operations in error correction level.

In order to improve annotation efficiency, we have developed a visual online tool to support the annotation procedure. In addition, we applied pattern matching and rule-based scripts to automatically convert a large amount (72.3%) of prompts into operation labels. We show the interface of our visual annotation tool in Appendix C.

As for annotation process, we hire 20 undergraduate students and 4 expert examiners to annotate and verify the GEC labels. We follow the annotation procedure in SuperGLUE (Wang et al., 2019a)

Type	Example
IWC	自己有双聪明能干的手，什么都能做出来。 You have smart hands to do everything. (Tips: “hands” cannot be combined with “smart”)
CM	绿色植物具有产生氧气。 Plants have (the ability) to produce oxygen. (Tips: Lack of object “the ability”)
CR	我们已走了约十里左右的路程。 We had walked about 10 miles or so. (Tips: “about” and “or so” are redundant)
SC	交通事故发生的原因是开车看手机造成的。 Traffic accidents are caused by (because) looking at cell phones while driving. (Tips: the structure of “because” and “caused by” cannot appear together in one sentence)
IWO	我改正并认识了自己的错误。 I corrected and realized my fault. (Tips: realize the fault first and correct it later)
ILL	我们应该防止事故不发生。 We should prevent accidents from not occurring. (Tips: double negation causes illogical errors)
AM	刚一开门，看病的就进来了。 As the door opened, the doctor/patient came in. (Tips: there is an ambiguity about who comes in)

Table 1: Examples of different types of errors.

that each annotator should work on test data first. After that, they can compare their labels with the gold ones. We encourage them to discuss their mistakes, questions and standards with other annotators and experts. To attain high-quality annotation with multiple references, we duplicate the sentences in our corpus 2 to 4 times. Furthermore, it is guaranteed that the redundant sentences are annotated by different annotators. Then experts are asked to review data that the annotators could not in agreement on the labels and add reasonable references. It is worth mentioning that we search for possible synonyms of the characters generated by *Insert* and *Modify* operations in annotation. We believe that supplying more word choices to annotators can improve the multi-reference rate. Moreover, we set up a weekly communication meeting to discuss common issues in annotation and adapt the labeling criteria. In total, the entire annotation procedure lasted for more than 4 months.

2.4 Quality Control

To ensure the high-quality of our FCGEC, we adopt the following five criteria: (1) Each sentence is in-

Corpus	Source	Paradigm	Sentence	#Error	#Refs	#Length
NLPCC(2018)	CFL	*Error-coded*	2000	1983(99.15%)	1.1	29.7
CGED	CFL	*Error-coded*	30145	25837(85.71%)	1.0	46.6
CTC-Qua(2021)	Native	*Error-coded*	972	482(49.59%)	1.0	48.9
MuCGEC(2022)	CFL	*Rewriting*	7063	6544(92.65%)	2.3	38.5
FCGEC (Ours)	**Native**	***Operation***	**41340**	**22517(54.47%)**	**1.7**	**53.1**

Table 2: The comparison of different Chinese grammatical error correction corpus. Numbers in row **#Error** mean the percentage of incorrect sentences in the corpus. **#Refs** indicates the average number of references contained in each sentence on average while **#Length** stands for the average number of characters in each sentence. Note that CGED is a combined corpus from 2016 to 2018 (Rao et al., 2018, 2020).

Subset	Sent.	Err.	#S	#D	#I	#M
Train	36340	19761	3930	10468	8705	7459
Valid	2000	1102	262	465	553	453
Test	3000	1654	421	1496	919	746

Table 3: Some statistics of FCGEC, including the number of sentences, the number of erroneous sentences, and the number of four operations (#S, #D, #I, #M denote *Switch*, *Delete*, *Insert*, *Modify*, respectively).

Figure 2: Correlation between types and operations.

spected by two specialized annotators to correct spelling and punctuation errors before annotation. Meanwhile, they have to eliminate the incomplete sentences (due to unexpected text truncation). (2) The specialized annotators were also asked to tag the sentences from news aggregator source that might have grammatical problems while checking spelling errors. Then these potential sentences will be discussed in weekly communication meeting. (3) We ask the annotators to read our guidelines and annotate twenty test instances. Then experts check their accuracy of the annotation. The annotators that meet the accuracy (90%) could continue to label the official data. (4) We assign 2x to 4x annotators per sentence for the corpus. In case their annotations are different, the experts will determine the correct labels. After that, annotators can also learn from these mistakes to achieve self-improvement. (5) After annotation, we unify the annotated labels under the minimal operation criteria inspired by Dahlmeier and Ng (2012) which applies fewer operations during correcting grammatical errors. More details about minimal operation algorithm is described in Appendix E.

2.5 Data Statistics and Comparison

We compare our corpus with other Chinese grammatical error datasets in Table 2. Moreover, the concrete statistics of FCGEC are shown in Table 3 and Appendix D. We summarize the advantages of our FCGEC in the following three aspects:

Multiple References. As discussed in Bryant and Ng (2015) and Zhang et al. (2022), the training and evaluation of GEC models can benefit from multiple references. In order to obtain more references, we ask the annotators to submit different reasonable operations for correcting errors. Meanwhile, we specifically provide several choices of synonyms for the generated text during annotation. We search for synonyms using both fine and coarse granularity. The fine-grained approach is to obtain synonyms from electronic dictionaries, while the coarse-grained way relies on similarity of the word vectors. It enhances the ratio of multiple references.

More Pragmatic Data. Pragmatic data involves errors in logic, common sense, ambiguity, etc. We increase the proportion of pragmatic data (Table 6) compared to other CGEC datasets, thus rendering the data more complex and challenging. Notably, we fix the ambiguity errors by providing references to correct them from different semantics.

Effective Error Types. We assign more refined error types to the grammatical errors, and these types are closely related to the correction operations. As shown in Figure 2, error types are always highly relevant to particular operations (e.g., CM

and CR rely on *Insert* and *Delete* operations respectively). We believe that error types can be utilized as auxiliary data to improve the performance of the GEC models to correct grammatical errors.

3 Benchmark Models

We divide the GEC task into a classification task and a correction task. The classification task involves the error detection and error type identification. We adopt the pre-trained language models (PLMs) based approaches for these tasks. As for the correction task, we propose a STG model to correct errors. Meanwhile, two categories of mainstream GEC models (Seq2Seq and Seq2Edit) are applied as benchmark models for our FCGEC.

3.1 Baselines for Classification Task

In error detection subtask, the model needs to determine whether a given sentence contains grammatical errors. Therefore, it is a binary classification task while the type identification subtask can be regarded as a multi-class classification task. The model should predict which of the seven error types the given erroneous sentences belong to. Note that some sentences may have multiple error types.

Recently, PLMs are proved to be effective and achieve success in various fields, such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), BERT-WWM (Cui et al., 2021), MacBERT (Cui et al., 2020) and StructBERT (Wang et al., 2019b). Therefore, we adopt different PLMs enhanced models as the benchmark models for these classification tasks. Specifically, we treat multiple PLMs as the backbone network and apply fully-connected layers on top of it for detection and identification.

3.2 Proposed STG Model

To correct grammatical errors, we propose an effective benchmark model, STG, which tackles error correction in the low-resource settings. Figure 3 gives an overview of our model. STG decomposes the error correction task into three processing modules: *Switch* , *Tagger* and *Generator*. The *Switch* module determines the order of characters that appear in the output on the basis of pointer network (Vinyals et al., 2015). Our *Tagger* module predicts the operation tags of each character and the number of characters that need to be generated in sequence. As for the *Generator* module, it fills in the characters that are not present in source sentence. Notably, each module can be trained independently.

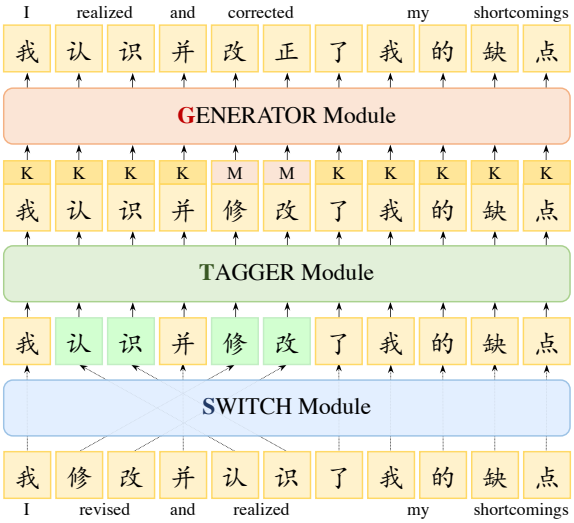

Figure 3: The architecture of our STG model.

Switch Module. The input to *Switch* module for character i is the hidden representation $h_i \in \mathbb{R}^d$ from PLM, where d denotes the dimension of hidden representations. Then *Switch* module determines the next position index for character i based on the pointer network. We apply the self-attention to predict which possible character $e(i)$ would be pointed to. It can be formulated as:

$$p\left(e(i)|h_i\right) = \text{attention}\left(h_i, h_{e(i)}\right) \quad (1)$$

The self-attention with scaled dot-product can be computed as below:

$$A = \text{Attention}\left(\boldsymbol{Q}, \boldsymbol{K}\right) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d}}\right) \quad (2)$$

where A is attention score matrices, $\boldsymbol{Q}$ and $\boldsymbol{K}$ are both linear projections of h. More details about our *Switch* module can be found in H.1.

Tagger Module. We first define five tags corresponding to the three operations (except *Switch* operation) as follows: the *KEEP* (*K*) tag is utilized to maintain the source character while the *DEL* (*D*) tag is assigned to remove character from source sequence. The tag of *INS_t* (*I_t*) represents the insertion of t words after the current character. The substituted character is marked as *MOD* (*M*) tag for *Modify* operation. As for the special case where the character is both substituted and required to insert other characters, we set the tag of *MINS_t* (*MI_t*) similar to *I_t*. Modification tags can perform a combination of multiple operations on a single character at the same time, thus eliminating the

need to correct the sentence via iterations as other edit-based methods. Limited by space, we provide some concrete examples in the Appendix H.2.

We take the prediction of tags and the number t of characters to be inserted or substituted for each character as classification task. Therefore, we apply two fully-connected layers to obtain tags and the number t. They can be written as:

$$T = \sigma\left(W h_i^s + b\right) \tag{3}$$

where T denotes the tag or number t while h_i^s is the hidden representations of character i. And σ stands for the $\mathrm{softmax}$ function. W and b are the learned weights and bias.

Generator Module. As we can leverage the masked language modeling (MLM) task (Devlin et al., 2018) of BERT-style PLMs for generating the characters which do not appear in the source sequence, *Generator* module inserts or substitutes the character with a certain number t of `[MASK]` token according to their tags. Then it predicts which characters are suitable to fill into the masked places.

Training and Testing. During the training process, we utilize cross-entropy loss $\mathcal{L}_{switch}$, $\mathcal{L}_{tag}$ and $\mathcal{L}_{gen}$ for the three modules. The STG model can be trained in two paradigms: *independent* and *joint*. The difference between them is whether each module is trained separately and thus they cannot utilize the shared encoder. We combine the loss in *joint* paradigm as follows:

$$\mathcal{L}(\theta) = \lambda_1 \mathcal{L}_{switch} + \lambda_2 \mathcal{L}_{tag} + \lambda_3 \mathcal{L}_{gen} + \gamma\lambda\|\Theta\|_2 \tag{4}$$

where $\lambda_\cdot$ is the coupling co-efficiency that regulates the three losses.Θ represents all trainable parameters in STG model and γ denotes the coefficient of L_2-regularization. $\mathcal{L}_{tag}$ is always larger than the other two losses, thus we generally set it one order of magnitude smaller. Furthermore, we train STG model with type identification (TTI) task to utilize auxiliary type data to improve model performance.

As for testing phase, we feed the erroneous sentence into each module in a pipeline fashion to correct errors. Specifically, we adopt constrained beam search to decode the sequence order.

3.3 Other Baselines for Correction Task

In order to present mainstream error correction models on our corpus, we take two categories of approaches as benchmark models:

Seq2Seq Models. A portion of the works adopt Transformer-based (Vaswani et al., 2017) encoder-decoder architecture as Seq2Seq fashion for correcting grammatical errors. The neural machine translation (NMT) based method is adopted in Fu et al. (2018) to tackle CGEC. Besides, Kaneko et al. (2020) utilize BERT-fuse to incorporate BERT into an encoder-decoder model for GEC. Meanwhile, MuCGEC (Zhang et al., 2022) presents a benchmark model based on Seq2Seq architecture with the Chinese BART (Shao et al., 2021).

Seq2Edit Models. Recent works also focus on the Seq2Edit models, which correct errors by labeling manipulations of each character. LaserTagger (Malmi et al., 2019) is applied to modify the sequence with three types of edits: *insertion*, *deletion* and *substitution*. PIE (Awasthi et al., 2019) presents iterative edit with custom inflection operations to correct the grammatical errors. GECToR (Omelianchuk et al., 2020) is an iterative sequence tagging framework with custom g-transformations that we adapt it to CGEC follow the efforts of Zhang et al. (2022).

4 Experimental Results

4.1 Evaluation Metrics

Classification Task. We regard the error detection task and error type identification task as classification tasks. Therefore, we adopt four common metrics, i.e., *Accuracy*, *Precision*, *Recall* and Macro F_1 *score* to evaluate the model performance.

Correction Task. As for correction task, we employ two different metrics : (1) *Exact Match* metric is obtained by calculating the percentage of corrected sentences for model outputs that exactly matched with the golden references. (2) The character-level edit metrics proposed by MuCGEC (Zhang et al., 2022) are utilized to compute fine-grained model performance. After obtaining the optimal sequence for character-level editing, they merge consecutive edits of the same type into span-level for both model outputs and golden references. Then MuCGEC calculates the highest *Precision*, *Recall* and $F_{0.5}$ *score* by comparing the edits of model outputs with each golden reference.

4.2 Experimental Settings

We conduct detailed experiments for fairly comparing benchmark approaches on our FCGEC. In classification tasks, we adopt officially released PLM

Model	Acc	P	R	F_1
• Grammatical Error Detection				
BERT	72.17	71.99	72.12	71.75
MacBERT	74.12	74.15	74.25	73.98
RoBERTa	74.84	74.82	74.91	74.68
MacBERT-Large	77.07	76.99	77.01	76.86
RoBERTa-Large	76.82	76.69	76.86	76.68
StructBERT-Large	**77.63**	**77.48**	**77.73**	**77.52**
Human	**82.65**	**84.08**	**83.88**	**82.63**
• Grammatical Error Type Identification				
BERT	56.53	56.87	63.15	59.15
MacBERT	53.51	56.20	64.58	59.22
RoBERTa	54.90	58.78	65.34	61.33
MacBERT-Large	57.86	57.29	63.51	59.76
RoBERTa-Large	**60.04**	59.86	**69.28**	**64.10**
StructBERT-Large	57.86	**60.13**	67.98	62.68
Human	**73.04**	**76.74**	**57.15**	**64.40**

Table 4: Average performance comparison on baselines among 10 independent runs for classification tasks.

parameters from HuggingFace website[1]. Then we fine-tune the different PLMs on our FCGEC for 4 epochs with batch size of 64. As for the correction task, we employ RoBERTa as the backbone PLM of our STG model and other benchmark models for training 100 epochs. Notably, the BART PLM (Lewis et al., 2019) utilized in Seq2Seq models is substituted by CPT (Shao et al., 2021). We set maximum t to 6 in *Tagger* module (It can cover 98% of the cases). In addition, we apply AdamW (Kingma and Ba, 2014) optimizer with a learning rate of 2e-5 and weight decay of 1e-2 for all tasks.

4.3 Human Evaluation Results

We hire 25 annotators from the crowd-sourcing platform of NetEase with a wide range of degrees and occupations. Specifically, annotators are restricted to be native speakers. We require them to annotate 10 instances for familiarization with the task requirements. Then they annotate 7,500 pieces of data (we randomly select 1,500 sentences from the test set and duplicate them 5 times), which is used to evaluate our human performance. As shown in Table 4 and 5, the error detection task is relatively easy for humans while error type identification task is harder due to the fact that it has more categories. As for correction task, it is also challenging for humans to correct grammatical errors. We further discuss the human performance based on the model performance in Section 4.4.

[1]https://huggingface.co/models

4.4 Overall Performance

The results of the classification tasks on FCGEC are shown in Table 4 for different PLMs from which several observations can be derived. First, the large-size variant PLMs perform better than other base-size models on the both detection and identification tasks as they can represent richer semantic information. To illustrate this observation, we roughly divide the error types into semantic and syntactic groups. We find that the average accuracy improvement for larger PLMs is significantly higher on the semantic group (7.6%) compared to the syntactic group (2.8%). Second, StructBERT-Large outperforms all PLMs on detection task while RoBERTa-Large achieves better performance on identification task, demonstrating two strong baselines at FCGEC. Moreover, there is an interesting observation on identification task that the humans have a lower performance of *Recall* than all PLMs, while the *Precision* is significantly better than them.

In terms of the correction task, Table 5 demonstrates the results of benchmark models on FCGEC. The overall performances of Seq2Edit-based models are better than the Seq2Seq-based models. Furthermore, our STG-series models substantially outperform them on FCGEC, which proves the effectiveness of STG architecture. Finally, there is still a significant gap comparing best-performing models with humans in all tasks. Moreover, the difficulty of the task also increases gradually on the detection, identification and correction, which are reflected on the difference of gap between models and humans.

4.5 Comparative Analysis

Independent training vs. Joint training. As we describe in Section 3.2, the *Switch*, *Tagger*, *Generator* modules in our STG can be trained flexibly either independently or jointly. In Table 5, STG with joint training (STG-Joint) brings gains of 4.17% in EM score and 4.86% in $F_{0.5}$ score compared with independent training STG (STG-Indep). The results indicate that the performance of correction can be enhanced during joint training since each module of STG can share more information and complement each other under a unified optimization objective.

Investigate the benefit of error type identification to correction. In Figure 2, we illustrate the correlation between error types and the operations of correction. We observe a significant association among error types and operations, which motivates

Model	EM	P	R	$F_{0.5}$
• **Seq2Seq Models**				
BERT-fuse	10.88	19.06	18.94	19.04
MuCGEC	21.16	39.47	26.19	35.84
• **Seq2Edit Models**				
PIE	22.07	29.15	29.77	29.27
LaserTagger	28.42	36.60	31.16	35.36
GECToR	15.66	31.06	18.74	27.45
• **Our Models**				
STG-Indep	29.93	43.16	32.88	40.62
STG-Indep+TTI	30.77	44.89	33.52	42.04
STG-Joint	**34.10**	**48.19**	**37.14**	**45.48**
Human	**65.25**	**79.46**	**67.57**	**76.76**

Table 5: Performance comparison for error correction tasks. Notably, EM indicates the metric of *Exact Match*.

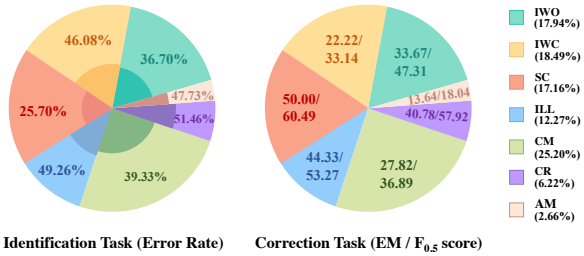

Figure 4: The fine-grained performance on identification and correction tasks. The numbers in the sectors of the pie chart indicate the error rate and EM / $F_{0.5}$ score on identification task and correction task, respectively.

us to treat error type identification as an auxiliary task (TTI) for training STG model. As shown in Table 5, STG-Indep+TTI indicates that the three modules of STG are trained independently with the TTI task incorporated. Our STG achieves better performance after integrating the TTI task compared with STG-Indep, which demonstrates the efficient error type data can be utilized as auxiliary data to enhance model correction performance. Moreover, STG-Indep+TTI can also obtain an accuracy improvement of 1.78 points on the error type identification task compared to RoBERTa.

Fine-grained performance analysis. In Figure 4, we demonstrate the fine-grained performance based on grammatical error types for identification task with RoBERTa-Large and correction task with STG-Joint. Notably, the dark sectors in the pie chart of the identification task indicate the proportion of errors for visual comparison. First, we observe the minimum error rate on SC, indicating that the PLM is more sensitive to syntactic structure errors. Second, the PLM performs weakly in terms of word-level errors, especially CR. After analyzing the error scenarios, we discover that the PLM may easily treat CR and CM errors as IWC errors. Furthermore, the PLM fails to determine the error types at the pragmatic level (i.e., ILL and AM), which illustrates the challenge of FCGEC.

As for correction task, it is clear that the performance on CM and IWC is inferior. We consider this potentially due to the fact that CM and IWC always require the generation of characters, increasing the difficulty of correction. Moreover, the model encounters trouble with AM due to the inclusion of pragmatic data such as ambiguity. It is hard for the model to distinguish the semantics in the sentences and correct it. In addition, we present more comparisons and fine-grained analyses in Appendix I.

Influence of the three modules in STG. The correction performance of our STG model is affected by three modules simultaneously. Thus we further investigate the impact of these modules on the STG-Joint. As shown in Figure 5, we analyze the char-level and sentence-level accuracy of each module. We ignore the *Keep* tag when calculating the char-level accuracy in *Tagger* module. *Tagger-t Acc.* is computed for the number t of *I_t* and *MI_t* tags. The first observation is that the performance of model is mainly constrained by *Tagger* module, while it fails to predict tags and number t precisely. Secondly, the performance of the *Generator* module illustrates that it is possible to achieve acceptable performance via only utilizing non-autoregressive approach with fine-tuning. Furthermore, despite the high performance of the *Switch* module, its role as the first module in the pipeline has a significant impact on the *Tagger* and *Generator* modules. Therefore a more robust performance of the *Switch* module is needed.

5 Related work

There already exists a lot of work on grammatical error correction for datasets and approaches. In terms of the dataset, most researches focus on English GEC. NUCLE (Dahlmeier et al., 2013), an early annotated corpus of GEC research, collects the erroneous sentences from students' essays in NUS. JFLEG (Napoles et al., 2017) is constructed from TOEFL exam with native sounding judgement. W&I (Bryant et al., 2019) collects the texts from non-native English students around the world in an online web platform and then manually an-

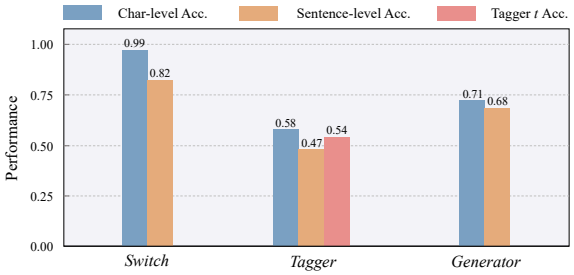

Figure 5: The performances of three modules in STG.

notates them for GEC. By contrast, the errors in LOCNESS (Bryant et al., 2019) are acquired from essays written by native English students. Unlike the previous dataset, CWEB (Flachs et al., 2020) focuses on grammatical errors in low error density domains from websites. However, the scale of the data is relatively insufficient in CGEC. NLPCC (Zhao et al., 2018), CGED (Rao et al., 2020), YACLC (Wang et al., 2021a) and MuCGEC (Zhang et al., 2022) are four publicly available non-native speaker resources for CGEC community, which encourage us to construct a high-quality CGEC corpus derived from native speakers.

As for the progress of GEC approaches, Seq2Seq and Seq2Edit are two mainstream approaches that achieve competitive results. Most of the work is based on Seq2Seq framework that generates the correct sentences directly (Zhou et al., 2019; Wan et al., 2020; Zhao and Wang, 2020; Kaneko et al., 2020). Furthermore, after Malmi et al. (2019) first apply the Seq2Edit approach, PIE (Awasthi et al., 2019) and GECToR (Zhang et al., 2022) are proposed to correct errors with iterating. After that, Tarnavskyi et al. (2022) employ an ensembling approach on GECToR for better performances.

6 Conclusion

In this paper, we construct a large-scale corpus for Chinese grammatical error detection, identification and correction. Compared to previous CGEC corpus, our FCGEC is more complicated and challenging with pragmatic data. Furthermore, we provide multiple references so that the models can be evaluated for better performance. Furthermore, we propose a STG model to correct grammatical errors. Extensive experiments demonstrate that our STG outperforms the baselines and achieves the state-of-the-art performance. However, experiments show that there exists a notable gap between cutting-edge models and humans. Therefore, it encourages the future GEC models to bridge the gap.

Limitations

The limitations of our work can be categorized into two main aspects: our corpus and model.

Limitations of FCGEC. In our FCGEC, a small number of sentences can be considered as different types of grammatical errors depending on correction methods. We do not provide a finer distinction between error types in this version. However, such fine-grained labels may supply more benefit in correction tasks (employing TTI as a auxiliary task).

Limitations of STG. The major limitation of our STG is that although no iteration is required, it corrects the errors via a pipeline paradigm with each modules in inference stage, thus it takes more time in the inference stage. Moreover, we consider that better performance may be achieved if the *Generator* module is pre-trained with a massively parallel corpus such as Lang-8 (Zhao et al., 2018), which we do not conduct in this paper.

Ethics Statement

Licensing Issues. FCGEC is a CGEC dataset collected from public examination websites and news aggregator sites. We collect the original grammatical error data or news data under the license of these sites or request for their permission. Meanwhile, the full attribution for original source of the data is cited in our FCGEC. In addition, we also commit not to use the corpus for commercial purposes, but only for the research studies.

Annotator Compensation. In our annotation procedure, we hire two categories of annotators. The first type is the annotators who annotate or examine the data for our FCGEC. We estimate that a skillful annotator requires about 1 to 2 minutes for each sentence to identify the error types and correct errors. On this basis, we pay the annotator 7 yuan (about 1 dollar) for 10 sentences. The second category is annotators from the crowd-sourcing platform of NetEase for computing human performances on FCGEC. Since they only require to do the measurement of the data, we set the compensation to 4 yuan (about 0.6 dollar) per 10 sentences.

Acknowledgements

This research is supported by the Science and Technology Project of Zhejiang Province (2022C01044) and the National Natural Science Foundation of China (51775496).

References

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *EMNLP-IJCNLP*.

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 and Hwee Tou Ng. 2015. How far are we from fully automatic high quality grammatical error correction? In *ACL*, pages 697–707.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. In *EMNLP*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *NAACL*, 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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Simon Flachs, Ophélie Lacroix, Helen Yannakoudakis, Marek Rei, and Anders Søgaard. 2020. Grammatical error correction in low error density domains: A new benchmark and analyses. In *EMNLP*.

Kai Fu, Jin Huang, and Yitao Duan. 2018. Youdao's winning solution to the nlpcc-2018 task 2 challenge: a neural machine translation approach to chinese grammatical error correction. In *NLPCC*, pages 341–350. Springer.

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 *ACL*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *ICLR*.

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. In *ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. Felix: Flexible text editing through tagging and insertion. In *EMNLP*.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *EMNLP*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *EACL*.

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 *CoNLL: Shared Task*, pages 1–14.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: tag, not rewrite. In *BEA@ACL*.

Gaoqi Rao, Qi Gong, Baolin Zhang, and Endong Xun. 2018. Overview of nlptea-2018 share task chinese grammatical error diagnosis. In *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 42–51.

Gaoqi Rao, Erhong Yang, and Baolin Zhang. 2020. Overview of nlptea-2020 shared task for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 25–35.

Alla Rozovskaya and Dan Roth. 2019. Grammar error correction in morphologically rich languages: The case of russian. *Transactions of the Association for Computational Linguistics*, 7:1–17.

Keisuke Sakaguchi, Courtney Napoles, Matt Post, and Joel Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. *TACL*, 4:169–182.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Maksym Tarnavskyi, Artem Chernodub, and Kostiantyn Omelianchuk. 2022. Ensembling and knowledge distilling of large sequence taggers for grammatical error correction. In *ACL*.

Viet Anh Trinh and Alla Rozovskaya. 2021. New dataset and strong baselines for the grammatical error correction of russian. In *Proceedings of ACL-IJCNLP*, pages 4103–4111.

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

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *Advances in neural information processing systems*, 28.

Zhaohong Wan, Xiaojun Wan, and Wenguang Wang. 2020. Improving grammatical error correction with data augmentation by editing latent representation. In *COLING*, pages 2202–2212.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Jiangnan Xia, Liwei Peng, and Luo Si. 2019b. Structbert: Incorporating language structures into pretraining for deep language understanding. In *ICLR*.

Yingying Wang, Cunliang Kong, Liner Yang, Yijun Wang, Xiaorong Lu, Renfen Hu, Shan He, Zhenghao Liu, Yun Chen, Erhong Yang, et al. 2021a. Yaclc: A chinese learner corpus with multidimensional annotation. *arXiv preprint arXiv:2112.15043*.

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

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *NAACL-HLT*, pages 380–386.

Yue Zhang, Zhenghua Li, Zuyi Bao, Jiacheng Li, Bo Zhang, Chen Li, Fei Huang, and Min Zhang. 2022. Mucgec: a multi-reference multi-source evaluation dataset for chinese grammatical error correction. In *NAACL-HLT*.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the nlpcc 2018 shared task: Grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing (NLPCC)*, pages 439–445. Springer.

Zewei Zhao and Houfeng Wang. 2020. Maskgec: improving neural grammatical error correction via dynamic masking. In *AAAI*, pages 1226–1233.

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

A Source of Data Collection

For the source of public examination websites, we collect the practice exercises from KS5U (http://5utk.ks5u.com/main.aspx), which are accessible online for public usage. As for the news aggregator sites (e.g., ZAKER, IT Home etc.), we randomly collect the titles or topic sentences from these sites. After that, we manually check and correct the sentences as we describe in Section 2.4 to ensure the quality of our corpus.

B Data Structure

We employ the JSON format to construct our FCGEC, as illustrated below:

```
{
  "id(The global id of the instance)": {
    "sentence": The original sentence,
    "error_flag": Whether sentence contains errors,
    "error_type": The error types of sentence,
    "operation" : [
      { The operation of the first reference },
      { The operation of the second reference },
      { ... }],
    "external" : Additional information
  }
}
```

For the format of four operations, we define each of the operations as follows:

» Suppose the given sentence is "A B C D E".

1. *Switch* operation

{"Switch":[0,2,1,3,4]} ("A B C D E" → "A C B D E")

// The values in the list indicate the order of the original character index after the swap (Index starts from 0).

2. *Delete* operation

{"Delete":[3]} ("A B C D E" → "A B C E")

// The characters indexed in the list will be deleted.

3. *Insert* operation

{"Insert":[{"pos":1,"tag":"INS_1","label":["F"]}]} ("A B C D E" → "A B F C D E")

// Insert a "F" after the character indexed by "pos".

4. *Modify* operation

{"Modify":[{"pos":2,"tag":"MOD_1","label":["F"]}]} ("A B C D E" → "A B F D E")

// Modify the character with index "pos" to "F".

C Visual Interface of Annotation Tool

As shown in Figure 6, we develop a visual annotation tool for the annotation process. Given an erroneous sentence, annotators are able to utilize this tool for easily identifying the error types and correcting the errors via four operations, i.e., *Switch*, *Delete*, *Insert* and *Modify*. Concretely, the annotators only need to drag or click on the small blocks which represent each character at the bottom of the main form to complete the corresponding operation. In order to support multiple references of the correction task, we provide a button for adding a reference up to five. Moreover, the small form on the right side displays correction prompts from teachers and experts to inform the annotators of the correction. Our tool can automatically convert these prompts to operations via rules. In addition, the real-time correction labels are displayed on the bottom of this small form.

Furthermore, our tool enables convenient comparison of multiple annotation operations of a sentence by different annotators, so that expert examiners can select the reasonable references. In practice, this flexible annotation tool has greatly accelerated our annotation procedure.

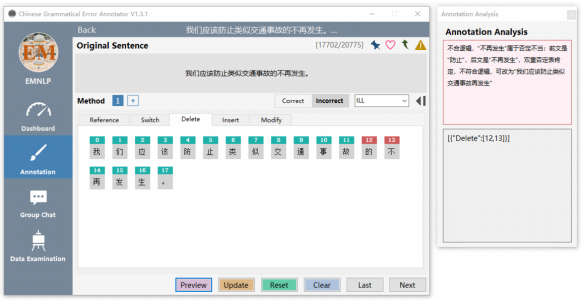

Figure 6: The screenshot of our annotation tool.

D More Statistics of FCGEC

We conduct more statistics on our corpus, including the length distribution of sentences, the proportion of error types in each category and the distribution of the number of reference numbers.

Length distribution of sentences. We compute the distribution of sentence lengths in our corpus. As shown in Figure 7, the length of the longest sentence is 359, while the shortest only contains 9 characters. Furthermore, the average sentence length is 53.06. Based on this, we can use various types of PLMs to encode sentences from the corpus without having to deal with exceeding the length.

Distribution of error types. In Table 6, we calculate the proportion of grammatical errors in the seven error categories on train set, validation set

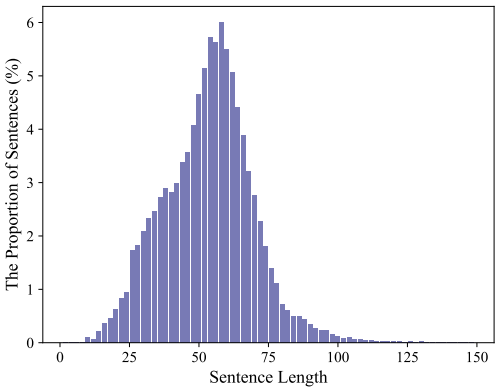

Figure 7: The length distribution of sentences in the whole corpus.

IWC	CM	CR	SC	IWO	ILL	AM
• **Train set**						
19.45	19.83	8.07	15.93	14.25	17.56	4.92
• **Validation set**						
21.58	20.71	6.41	19.41	16.90	11.27	3.73
• **Test set**						
18.50	25.21	6.23	17.17	17.96	12.27	2.66

Table 6: The proportion (%) of error types in Train, Validation and Test set, respectively.

and test set. We can observe that the error types are split as closely as possible with a similar distribution. Moreover, we can notice that the pragmatic type of error (ILL, AM and part of the IWO) also accounts for a significant proportion. Therefore our corpus tends to be more challenging.

The distribution of references. For the erroneous sentences, we allow the annotators to correct them through a variety of references. Thus the distribution of sentences with respect to the number of references is shown in Figure 8. First, we analyze sentences with a single reference and find that most of them are due to two scenarios: (1) The sentence is short or the grammatical error is very simple and obvious. (2) Some categories of errors often have only one way to correct them, such as IWO and ILL. Second, for sentences that contain two references, the errors of SC play a significant role in them. This is due to the fact that SC errors are always corrected by removing one of the two similar grammatical structures. More references for correcting sentences often indicate the need to insert or modify characters. Furthermore, we consider that the number of references could be increased via an enhanced and more refined annotation process and

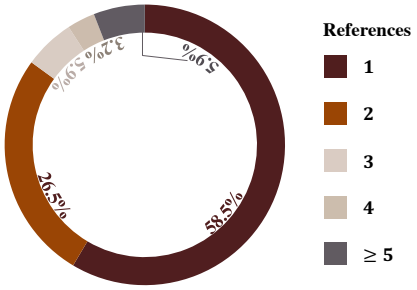

Figure 8: The distribution of the sentences in terms of references in our FCGEC.

Attribute	Value
Gender (Male / Female)	9 / 16
Age	[18, 54]
Native Language	Chinese
Education	Middle School Student
	Specialty
	Undergraduate Student
	Bachelor's Degree
	Master's Degree
Occupation	Students
	Freelancers
	Factory Workers
	Federal Employees
	Professionals (e.g., Lawyers)
	Office Staffs (e.g., Managers)

Table 7: The demographic information of annotators.

by assigning more annotators to each sentence.

E Algorithm of Minimal Edit

Given a pair containing the incorrect sentence and the corrected sentence, we design an algorithm to generate the operation labels under the minimal operation criteria from such pair. The algorithm is illustrated in Algorithm 1. It is mainly utilized in the following two scenarios: (1) Automatically convert the prompts of teachers and experts into operation labels for our visual annotation tool. (2) Based on this algorithm, we can unify the operation labels after annotation procedure to ensure that fewer operations are adopted during correcting grammatical errors for quality control. Besides, this can also help us to check for mistakes and guarantee the consistency of the data in the annotation.

In addition, we can convert the data from other formats (i.e., *rewriting* and *error-coded* paradigm) to our operation labels through this algorithm. Thus we can apply our STG model to other datasets.

Algorithm 1 Attain the operation labels via minimum edit distance.

Input: Source sentence S and target sentence T.
Output: The operation labels L that convert S to T.

if S == T **then**
  return [{}]
end if
$L = [\{\}]$
$tags$ = [“Copy”, “Modify”, “Delete”, “Insert”]
$mov = [(-1,-1),(-1,-1),(-1,0),(0,-1)]$
Calculate the character frequency f_S of S, and f_T of T.
if $f_S == f_T$ **then**
  Calculate the longest common substring s_1 and second longest one s_2 between S and T.
  Swap the positions of s_1 and s_2 with their indexes p_{ori} in S. Then the swapped sentence S_{swap} and indexes p_{swap} can be obtained.
  if $S_{swap} == T$ **then**
    $L = [\{\text{“Switch”} : p_{swap}\}]$
  else
    goto 17
  end if
else
  Calculate the Levenshtein distance and then obtain the edit path matrix M_p.
  $ops = []$
  getOperations(M_p, len(S), len(T), ops)
  Merge labels with adjacent index and the same operation in ops to get L
end if
return L

Function getOperations(M_p, i, j, ops)
if $i == 0$ and $j == 0$ **then**
  return
end if
for all op such that $op \in tags$ **do**
  if $M_p[i][j].get(op)$ **then**
    ops.append($[i, j, op]$)
    getOperations($M_p, i + mov[0], j + mov[1], ops$)
    break
  end if
end for
EndFunction

F The Demographic of Humans

In order to evaluate human performance on our FCGEC, we employ 25 annotators from the crowdsourcing platform of NetEase. Moreover, with the aim of measuring human performance as completely as possible, we hire a diverse range of annotators with different aspects (Education, occupation, age, etc.). As shown in the Table 7, the platform of NetEase provides us with their non-private demographic information about the annotators.

It is worth mentioning that we ask the annotators to label more data (randomly sampling 50% of the test set and then duplicating them 5 times) compared to other work as a way to attain more precise human performance.

G Details of Pre-trained Language Models

We enhance the performance of the model with PLMs for both the classification and correction tasks. In order to enable better reproduction of our results, we provide the details and links to officially released pre-trained parameters in the Table 8.

In Seq2Seq models, we adopt CPT as the Chinese BART model. In particular, since some Chinese punctuation is missing in the vocabulary of the BART model (e.g., Chinese quotation marks), we avoid performance degradation by substituting the punctuation with their English counterparts during pre-processing stage.

H Details of Our STG Model

To better illustrate the details of our STG model, we present additional input samples and the processing for the three modules in this section.

H.1 Switch Module

As we describe in Appendix B, the labels of *Switch* indicate the order of the original character index after swapping. However, the index of the next character is predicted for each character in our *Switch* module. Therefore, we need to fill this gap by converting these labels. We demonstrate the differences between these two label types in Figure 9.

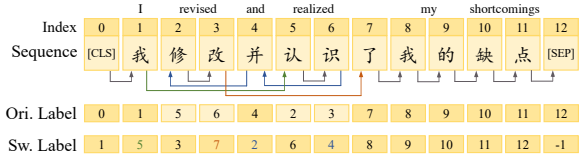

Figure 9: Comparison of the differences in the two types of *Switch* labels. **Ori. Label** and **Sw. Label** indicate the annotated labels and the processed labels in *Switch* module, respectively.

In *Switch* module, we utilize pointer network with self-attention mechanism to predict which character will be pointed to of each character. Furthermore, we adopt cross-entropy as the loss function to measure the margin between attention score matrix A and the golden labels for optimizing.

H.2 Tagger Module

In section 3.2, we introduce five tags (*K*, *D*, *INS_t*, *M* and *MI_t*) that correspond to the three operations (i.e., *Delete*, *Insert* and *Modify*) in *Tagger* module. For better understand our tags, we present some concrete examples in Figure 10.

PLM Name	Parameters	Url	Model Size
• PLMs of Basic Size			
BERT (Devlin et al., 2018) → BERT-wwm-ext RoBERTa† (Liu et al., 2019) → RoBERTa-wwm-ext	12 layers 768-d hidden Size 12 attention heads	https://github.com/ymcui/Chinese-BERT-wwm	102M
MACBERT (Cui et al., 2020) → MacBERT-base		https://github.com/ymcui/MacBERT	
CPT† (Shao et al., 2021) → CPT-base	10 layers of encoder 2 layers of decoder 768-d hidden Size 12 attention heads	https://github.com/fastnlp/CPT	116M
• PLMs of Large Size			
RoBERTa (Liu et al., 2019) → RoBERTa-wwm-ext-large	24 layers 1024-d hidden Size 16 attention heads	https://github.com/ymcui/Chinese-BERT-wwm	325M
MACBERT (Cui et al., 2020) → MacBERT-large		https://github.com/ymcui/MacBERT	
StructBERT (Wang et al., 2019b) → Structbert.ch.large		https://github.com/alibaba/AliceMind/tree/main/StructBERT	

Table 8: Detailed information of PLMs. Models with mark † are utilized as the backbone in correction task. The names after → are the specific version of the PLM models.

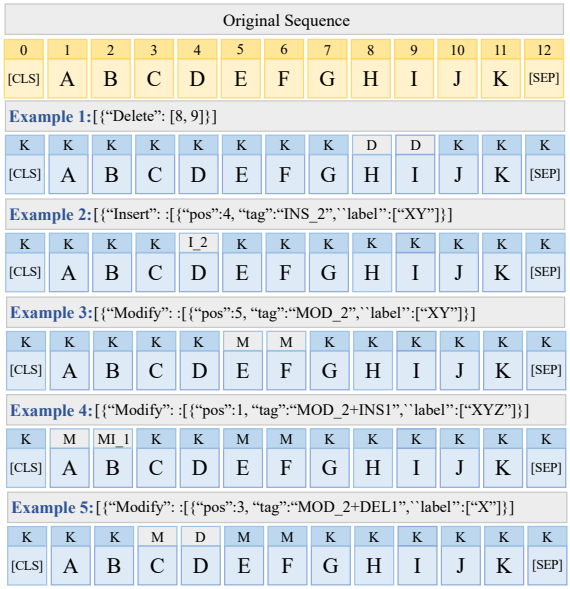

Figure 10: Examples of the tags for five typical cases in *Tagger* module.

With this well-designed tagging criterion, our STG model can perform arbitrary manipulations of the sequence without iteration. During the training stage, we employ two classification layers to determine the tags and the number t of *INS_t* and *MI_t*, separately. Meanwhile, The cross-entropy is also applied to compute the loss. We optimize both of the parameters in the two classification layers simultaneously by combining the loss of them.

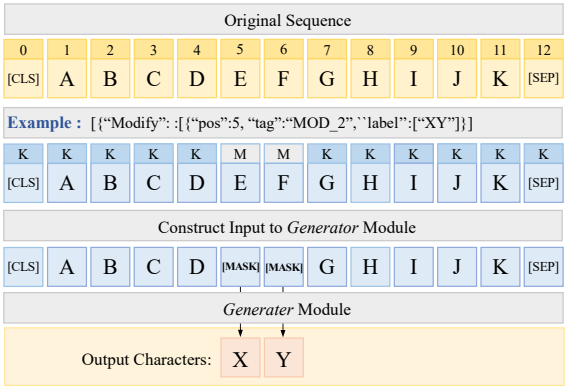

Figure 11: An example about the model input of *generator* module.

In particular, we try a small trick (Note that the trick is not adopted for the results of our STG models in Table 5, for fair comparison) that can further improve the performance of the model, which is to utilize the weighted cross-entropy loss. As the majority of tags in a sentence are *Keep*, it is intuitive to increase the weight of other tags to solve this typical category imbalance problem. After we conduct experiment on STG-Joint with this trick, we observe a 0.72% performance improvement in *Exact Match*.

H.3 Generator Module

In *Generator* module, we exploit the features of BERT-style PLMs to predict the characters which

Model	Switch		Delete		Insert		Modify	
	EM	$F_{0.5}$	EM	$F_{0.5}$	EM	$F_{0.5}$	EM	$F_{0.5}$
Ratio (%)	20.90		34.42		27.32		17.35	
Seq2Seq (MuCGEC)	12.33	20.85	31.45	54.42	10.90	18.57	8.58	23.56
Seq2Edit (LaserTagger)	16.16	20.51	40.43	**59.62**	7.34	13.23	15.18	28.14
STG-Joint (Ours)	**35.62**	**48.94**	**42.60**	59.29	**12.79**	**18.99**	**19.14**	**34.15**

Table 9: The metric (%) of *Exact Match* and $F_{0.5}$ *score* for each operation on the subset of test set. The first row (i.e., Ratio) represents the proportion of each operation to the total.

do not appear in the source sentence. The outputs of *Tagger* module are utilized to generate the input sequence with `[MASK]` tokens. We present an example of the input sequence in Figure 11.

After that, the *Generator* module predicts the indexes of the characters in vocabulary list that should be filled in at `[MASK]`. Similarly, the cross-entropy loss is adopted for optimizing the parameters in *Generator* module.

I More Comparisons and Analysis

To further investigate the differences in the ability of the correction models, we present more comparisons and analyses in this section.

Performance on four operations of correction. Table 9 illustrates another perspective of the fine-grained results that we calculate the performance of correction models on different operations. More specifically, we split the entire test set into four small subsets that contain only one operation for incorrect sentences. Then we compare our STG-Joint model with the best performing models in the Seq2Seq and Seq2Edit categories, respectively.

First, we can observe that our STG-Joint is significantly outperforms the other two models in terms of *Switch* operation. This is due to the fact that we design a special *Switch* module to efficiently handle such operation. Secondly, the performance of STG-Joint and LaserTagger is comparable in terms of *Delete* operation. However, Seq2Seq model behaves relatively weakly, due to its arbitrary modifications that often tend to delete more characters. Lastly, we discover that all models have poor performances on *Insert* and *Modify* operations, indicating that the task of generating new characters is more challenging.

Original performance of MuCGEC. In Section 4.4, we substitute the original backbone of the Seq2Seq and Seq2Edit models in MuCGEC (Zhang et al., 2022) for a fair comparison. We conduct additional experiments to demonstrate the original performance of models in MuCGEC. They employ the PLMs of Chinese-BART-Large[2] and StructBERT-Large[3] for Seq2Seq and Seq2Edit (GECToR) model, respectively.

Model	P	R	$F_{0.5}$
Seq2Seq+CPT-B	39.47	26.19	35.84
Seq2Seq+BART-L	38.59	36.52	38.16
Seq2Edit+RoBERTa-B	31.06	18.74	27.45
Seq2Edit+StructBERT-L	30.68	21.65	28.32

Table 10: Performance comparison for different PLM backbones for models in MuCGEC. The suffix of B represents base size, while L stands for large size.

We present the results of MuCGEC in Table 10. It is clear that the performances of models with large-size PLMs are better than those of the base-size PLMs. Specifically, Seq2Seq model is greatly improved after applying BART-Large as the backbone. Moreover, it is close to the results of our STG-series models. However, there is only a slight improvement of $F_{0.5}$ for the Seq2Edit model. After we further observe the error examples of Seq2Seq and Seq2Edit on the test set, we discover that the results of *Switch* operation are the critical limitation for Seq2Edit model. This also illustrates the necessity of the *Switch* module in our STG.

Performances on the validation set. In order to evaluate the distribution of our split dataset, we show the results of the best performing models on the corresponding validation set for the three tasks (detection, identification and correction) in Table 11. It is reasonable to observe that the performance on the validation set is slightly better than on the test set. Therefore, we keep the data distribution on the validation set close to the test set,

[2]https://huggingface.co/fnlp/bart-large-chinese

[3]https://github.com/alibaba/AliceMind/tree/main/StructBERT

Model	Acc	P	R	F_1
• **Grammatical Error Detection**				
StructBERT-Large	80.40	80.24	78.55	80.24
• **Grammatical Error Type Identification**				
RoBERTa-Large	60.44	68.17	64.11	65.77
Model	**EM**	**P**	**R**	$F_{0.5}$
• **Grammatical Error Correction**				
STG-Joint	36.71	50.00	39.21	47.39

Table 11: Corresponding validation performance for reported test result of the best performing models on three tasks.

which facilitates the model to search for the best hyperparameters on the validation set. Thus we can obtain better performance on the test set.

Computing resources and times. In Table 12, we show the detailed computing resources and hyperparameters for training our STG-Joint model. Meanwhile, we also record the training time consumed under these hyperparameters and devices.

Configuration	Value
Device	1 GeForce RTX 3090 (24G RAM)
PLM model	RoBERTa-Base-wwm-ext (Liu et al., 2019)
Number of epochs	100
Batch size	32
Beam size	5 / [1, 5, 10, 20]
Sequence Length	150 / [50, 100, 150]
Learning Rate	1e-5 / [5e-6, 1e-5, 2e-5, 5e-5]
Optimizer	Adam
Dropout	0.1
Weight Decay	1e-2
Total training time	About 12 hours
Hyperparameters Selection	Best performance on the validation set with the minimum loss

Table 12: Computing device, hyperparamters and training time for STG-Joint model. For hyperparameters of beam size, sequence length and learning rate, the left side of / is the best hyperparameter while the right side is the set for hyperparameter search.

J Case Study

In order to explore the performance of the models on the correction task, we conduct analyses on case examples. Similarly, we compare STG-Joint with the best performing models in the Seq2Seq and Seq2Edit categories (MuCGEC and LaserTagger). We present the cases in Table 13. Meanwhile, the English version of the cases can be seen in Table 14. In particular, since the ground truth contains multiple references, we represent one of them as an illustration due to space constraints. Note that the results of the models in Table 13 are based on the multiple references.

We can derive several observations from these case examples. First, in the category of word order errors (IWO), both Seq2Seq and Seq2Edit models can correct the elementary errors (Example 10). However, they fail to solve the more difficult order errors (progressive relationship problem in Example 9). Since our STG model is specifically equipped with the *Switch* module, it is possible to correct for these errors. Secondly, in the case of error categories that require the generation of new characters (e.g. CM and IWC), more improvements are required for all models. Finally, the pragmatic errors are the most difficult to correct (especially for AM). We encourage future models to pay more attention to these types of errors.

Type	Erroneous Sentence	Ground Truth	STG-Joint	Seq2Seq	Seq2Edit
IWC	**1.** 近些年来，我国全力做好癌症筛查、临床治疗和药品供应工作，努力减少癌症死亡率。	近些年来，我国全力做好癌症筛查、临床治疗和药品供应工作，努力降低癌症死亡率。	✔	✗	✔
	2. 我们有吃苦耐劳的人民，又有优裕的自然资源。	我们有吃苦耐劳的人民，又有丰富的自然资源。	✗	✗	✗
CM	**3.** 笔记本电脑充分显示了快捷、稳定、方便而成为各种赛事新闻报道的重要工具。	笔记本电脑充分显示了快捷、稳定、方便的特点而成为各种赛事新闻报道的重要工具。	✔	✗	✗
	4. 我们要养成爱读书，特别是读经典，读名著，让书香溢满校园。	我们要养成爱读书，特别是读经典，读名著的习惯，让书香溢满校园。	✗	✗	✗
CR	**5.** 为精简字数，这篇文章不得不略加删改一些。	为精简字数，这篇文章不得不略加删改。	✔	✔	✔
	6. 我们发自内心由衷地感谢老师多年来的默默付出。	我们发自内心地感谢老师多年来的默默付出。	✔	✗	✗
SC	**7.** 一个人变好变坏，关键在于内因起决定作用。	一个人变好变坏，内因起决定作用。	✔	✔	✔
	8. 期末考试前出现失眠、烦躁等现象，这往往是因为太在乎考试成绩，心理负担过重造成的。	期末考试前出现失眠、烦躁等现象，这往往是太在乎考试成绩，心理负担过重造成的。	✔	✗	✗
IWO	**9.** 参加奥运的选手们十分清楚，一场比赛的输赢，不仅关系到祖国的荣誉，而且关系到个人的尊严。	参加奥运的选手们十分清楚，一场比赛的输赢，不仅关系到个人的尊严，而且关系到祖国的荣誉。	✔	✗	✗
	10. 学校自从开展研究性学习以来，同学们踊跃参与，创新意识和创新能力得到很大的提升。	自从学校开展研究性学习以来，同学们踊跃参与，创新意识和创新能力得到很大的提升。	✔	✔	✔
ILL	**11.** “十一”放假之前，老师反复强调要防止不发生事故。	“十一”放假之前，老师反复强调要防止发生事故。	✔	✔	✔
	12. 面对一件棘手的事情，我们需要三思而后行。多动脑子，会避免少捅娄子少出错。	面对一件棘手的事情，我们需要三思而后行。多动脑子，会少捅娄子少出错。	✗	✗	✗
AM	**13.** 很多人认为科学家终日埋头搞科研，不问家事，有点儿不近人情，然而事实却是对这种偏见的最好说明。	很多人认为科学家终日埋头搞科研，不问家事，有点儿不近人情，然而事实却是对这种偏见的最好反驳。	✔	✗	✗
	14. 他决定背着妈妈去医院检查身体。	他决定瞒着妈妈去医院检查身体。	✗	✗	✗
	15. 张义和王强上课说话，被老师叫去办公室了。	张义和王强上课说话，两人被老师叫去办公室了。	✗	✗	✗

Table 13: The case study for comparing the performances of models. The characters in red denote the differences between erroneous sentences and ground truth. We demonstrate the English version in Table 14.

Type	Erroneous Sentence	Ground Truth	Tips
IWC	**1.** In recent years, we have struggled to cut down cancer mortality by improving cancer screening, clinical care and the supply of drugs.	In recent years, we have struggled to reduce cancer mortality by improving cancer screening, clinical care and the supply of drugs.	In general, "cut down" cannot be collocated with "cancer mortality", and "reduce" should be used.
	2. We have industrious people and favourable natural resources.	We have industrious people and abundant natural resources.	Usually, we pair "abundant" with "resources".
CM	**3.** Laptops demonstrate the speed, stability and convenience and become an important tool for news coverage of various events.	Laptops demonstrate the characteristics of speed, stability and convenience and become an important tool for news coverage of various events.	In Chinese, the incorrect sentence is missing the object "characteristics".
	4. We should develop of reading, especially reading classics and masterpieces, so that the fragrance of books can overflow the campus.	We should develop the habit of reading, especially reading classics and masterpieces, so that the fragrance of books can overflow the campus.	Similarly, the erroneous sentence misses the object "habbit" in Chinese.
CR	**5.** In order to reduce the word count, this article had to be slightly redacted some.	In order to reduce the word count, this article had to be slightly redacted.	The word "some" is redundant and can be deleted.
	6. From the bottom of our hearts, we thank our teachers sincerely for their quiet dedication over the years.	From the bottom of our hearts, we thank our teachers for their quiet dedication over the years.	In Chinese, the word "sincerely" is superfluous and should be removed.
SC	**7.** The crucial thing for a person to become good or bad is that the inner reasons play a role in determining it.	For a person to become good or bad, the inner reasons play a role in determining it.	The structure of "the crucial thing for" and "play a role in" are confusing.
	8. Insomnia and irritability before exams are caused by due to the excessive psychological burden with caring too much about exam results.	Insomnia and irritability before exams are caused by the excessive psychological burden with caring too much about exam results.	We can not apply the structure of "caused by" and "due to" in a sentence simultaneously.
IWO	**9.** Olympic athletes understand that winning or losing a race is not only about the honor of the country, but also about the dignity of themselves.	Olympic athletes understand that winning or losing a race is not only about the dignity of themselves, but also about the honor of the country.	"Honor of the country" and "dignity of themselves" should be swapped due to progressive relationship.
	10. The school since introduced the research study, students have participated enthusiastically and their creative awareness and ability have been greatly enhanced.	Since the school introduced the research study, students have participated enthusiastically and their creative awareness and ability have been greatly enhanced.	"Since" should be placed at the beginning of the sentence
ILL	**11.** Before the holiday, teachers emphasized over and over again to prevent accidents from not happening.	Before the holiday, teachers emphasized over and over again to prevent accidents from happening.	The double negation causes logical errors.
	12. When faced with difficulties, we need to think first. More thinking will avoid less troubles and mistakes.	When faced with difficulties, we need to think first. More thinking can lead to less troubles and mistakes.	More thinking leads to less errors in commonsense, while "avoid" causes errors.
AM	**13.** Many people think that scientists engage in research and do not communicate with others, yet the facts are the best illustration of this prejudice.	Many people think that scientists engage in research and do not communicate with others, yet the facts are the best rebuttal to this prejudice.	The semantic meaning is rather ambiguous that we cannot infer the role of facts on prejudice.
	14. He decided to carry (not to tell) his mother (he was going) to the hospital.	He decided not to tell his mother he was going to the hospital.	There is an ambiguity in Chinese.
	15. Yi Zhang and Qiang Wang talked in class and was called to the office by the teacher.	Yi Zhang and Qiang Wang talked in class and the two were called to the office by the teacher.	There is ambiguity on who was called to the office.

Table 14: The English version of erroneous sentences and ground truth in case study. Furthermore, we provide tips for better understanding the grammatical errors in Chinese.