CxGGEC: Construction-Guided Grammatical Error Correction

Yayu Cao[1], Tianxiang Wang[1], Lvxiaowei Xu[1], Zhenyao Wang[2], Ming Cai[1]*
[1]Department of Computer Science and Technology, Zhejiang University,
[2]Department of Computer Science and Technology, Dalian University of Technology
[1]{yayu_cao, wang_tx, xlxw, cm}@zju.edu.cn,
[2]oriyuquan@gmail.com

Abstract

The grammatical error correction (GEC) task aims to detect and correct grammatical errors in text to enhance its accuracy and readability. Current GEC methods primarily rely on grammatical labels for syntactic information, often overlooking the inherent usage patterns of language. In this work, we explore the potential of construction grammar (CxG) to improve GEC by leveraging constructions to capture underlying language patterns and guide corrections. We first establish a comprehensive construction inventory from corpora. Next, we introduce a construction prediction model that identifies potential constructions in ungrammatical sentences using a noise-tolerant language model. Finally, we train a CxGGEC model on construction-masked parallel data, which performs GEC by decoding construction tokens into their original forms and correcting erroneous tokens. Extensive experiments on English and Chinese GEC benchmarks demonstrate the effectiveness of our approach.

1 Introduction

Grammatical Error Correction (GEC) is the task of automatically detecting and correcting errors in text (Bryant et al., 2023), which after the advent of Transformer (Vaswani et al., 2017), has been categorized into two main types: Seq2Edit method and Seq2Seq method (Sun et al., 2021; Zhang et al., 2022b).

Seq2Edit method typically involves converting source sentences into a sequence of edit operations (Stahlberg and Kumar, 2020; Omelianchuk et al., 2020), which offers specific advantages in the GEC task due to its higher inference efficiency, while limited to manually selecting dictionaries (Awasthi et al., 2019; Malmi et al., 2019). Seq2Seq method treats GEC as a monolingual translation problem (Junczys-Dowmunt et al., 2018a; Sun et al., 2021) and demonstrates a better correction ability. Recent advances have enabled language models (LMs) to more adequately capture syntactic phenomena (Jawahar et al., 2019; Wei et al., 2022), making them capable GEC systems when little or no data is available (Zhang et al., 2022b; Wan and Wan, 2021). However, because the use of syntactic information of prior works is limited to the application of grammatical labels, we observe that currently no method can fully leverage the syntactic information and semantic usage patterns inherent to perform the GEC task.

Construction Grammar (CxG) (Goldberg, 1995, 2003) regards constructions (i.e., form-meaning pairs) as the fundamental units of linguistic knowledge, with each construction modeled as a sequence of slot-constraints (Dunn, 2017) which is composed of lexical items or syntactic labels. For example, "Subject-Verb–Object1–Object2" is a ditransitive construction (Goldberg, 1995) that represents the abstract meaning of transfer, while the modality construction "NOUN-AUX-be" expresses advice or suggestion (Xu et al., 2023). CxG claims that our knowledge of language is captured by network of constructions (Goldberg, 2003). Grammatical errors stem from a lack of sufficient knowledge about language usage (Bryant et al., 2023), making constructions beneficial for enhancing the GEC task. Some examples are demonstrated in Table 1, which shows the improvements of the GEC task by identifying potential constructions in the sentence.

Based on the above observation, we propose the following technical approach: (1) establishing a construction inventory from corpora, (2) identifying constructions from ungrammatical sentences, and (3) training models using ungrammatical sentences augmented with constructions for the GEC task.

However, realizing the above approach presents the following three challenges:

*Corresponding author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6143–6156
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

Ungrammatical Sentence	Identified Construction	Corrected Sentence
The book which I bought it yesterday is very interesting.	DET–NOUN–PRON–SUBJ–VERB	The book which I bought yesterday is very interesting.
The students in the library preparing for their exams.	DET–NOUN–ADP–DET–NOUN–AUX	The students in the library are preparing for their exams.
Some important departments need strict administration of their members.	VBP–ADJ–NOUN–ADP	Some important departments need strict administration for their members.

Table 1: Examples of three error types demonstrating the improvement of the GEC task using CxG: unnecessary, missing, replacement. (DET, NOUN, PRON, SUBJ, VERB, ADP, AUX, ADJ, and VBP denote determiner, noun, pronoun, subject, verb, preposition, auxiliary verb, adjective, and non-3rd person singular present verb, respectively.)

(Q1) What types of constructions are most effective in improving the performance of the GEC task?

(Q2) How can constructions be identified from ungrammatical sentences?

(Q3) How can the identified constructions be effectively utilized to guide the GEC task?

As for (Q1), an observation is that the guiding effectiveness of constructions is maximized when they overlap with or are adjacent to grammatical errors in sentences. Current methods for construction extraction can be categorized into manual extraction and automatic extraction (Xu et al., 2023). Manual extraction is limited by scale. Two primary automatic methods exist: one calculates bidirectional association scores between adjacent words (Dunn, 2017), while the other, CxGLearner (Xu et al., 2024), leverages LM token prediction probabilities. The former produces shorter constructions with limited structural completeness due to adjacent calculation method, whereas the latter, using LMs, generates more complete constructions with well-distributed lengths because it allows extended distances when assessing slot constraints. Thus, we adopt CxGLearner for constructing the construction inventory.

Regarding (Q2), current construction generation methods are only applicable to grammatical sentences. Inspired by Jiang et al. (2021) that LMs are insensitive to subtle differences between sequences, which means LMs exibit a certain degree of tolerance toward noise, we propose a LM-based approach to identify expected constructions from ungrammatical sentences.

To answer (Q3), we train a CxGGEC model based on a construction-augmented vocabulary. Through concatenating ungrammatical sentences with responding construction-masked sentences, CxGGEC is able to decode constructions into correct tokens by the Seq2Seq method.

Extensive experiments have been conducted to illustrate the superiority of CxGGEC on the GEC task, while multilingual experiments further indicate construction is beneficial across languages.

2 System Overview

Our CxGGEC framework can be divided into three steps: (1) construction generation, (2) construction masking, (3) CxG-guided GEC. Figure 1 displays the entire framework.

2.1 Construction Generation

Construction Inventory Establishment. Construction is represented as a sequence of slot-constraints. We annotate the part-of-speech tags in corpus from various domains, and employ CxGLearner (Xu et al., 2024) to extract constructions from annotated corpus, which assesses the association strength among slots based on LM. Therefore, we establish a well-distributed construction inventory, which will be taken as a construction vocabulary in subsequent training phase. The details of establishment are shown in Appendix D.

Identifying Construction in Ungrammatical Sentences. Because ungrammatical sentences may damage constructions, the construction inventory we obtained cannot be applied to identify expected constructions from ungrammatical sentences. Therefore, based on the tolerance of LMs for noise, we leverage the construction inventory to train a construction prediction model to identify constructions from ungrammatical sentences. The training details of the prediction model are demonstrated in Section 3.

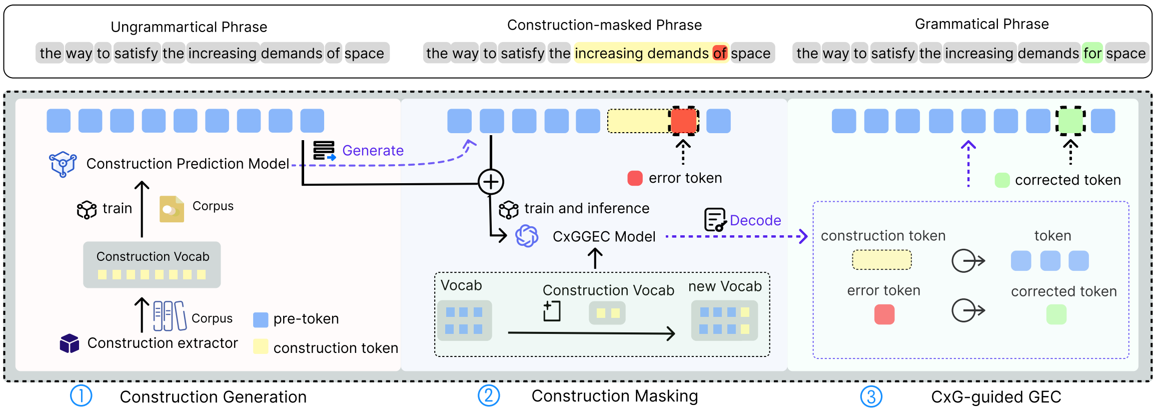

Figure 1: Overview of the proposed CxGGEC framework.

2.2 Construction Masking

To guide the GEC task with CxG, firstly we identify expected constructions from ungrammatical sentences through the construction prediction model, and then we construct the parallel training corpus by concatenation of the ungrammatical sentences with their construction-masked counterparts and the corresponding ground-truth sentences. Finally, we train a CxGGEC model by the Seq2Seq method.

2.3 CxG-guided GEC

For inference process, we concatenate the ungrammatical sentences with their construction-masked versions, forming a combined input just as the training phase. Specifically, construction masking serves as a context-aware signal that directs the model to locate parts requiring correction and output grammatical sentences by decoding construction tokens into original tokens and decoding error tokens into correct tokens. Through this construction-guided approach, the model aligns the grammatical error with the language usage patterns inherent in constructions, thereby improving the effects on GEC tasks.

3 Model

3.1 Construction Prediction Model

Construction Selection Strategy. Since constructions are often stored redundantly at different levels of abstractness, overlapping constructions can be captured by the grammar induction algorithm (Dunn, 2017, 2019). Xu et al. (2024) summarize the phenomenon of overlap into two scenarios: *Inclusion* and *Intersection*, which can lead to issues like redundancy and imbalanced encoding.

Based on our Seq2Seq training approach, it is essential to ensure that the constructions used to mask within the training sentences do not exhibit overlap or intersection. Drawing inspiration from RoBERTa's (Liu, 2019) dynamic masking approach, we randomly retain the overlapping sections for each sentence, while keeping the other parts intact. This method prevents overlaps and allows the model to learn diverse combinations of constructions, helping to mitigate the risk of the construction prediction model overfitting to specific construction patterns. The algorithm is depicted in Algorithm 1. CHECKOVERLAP($\cdot$) inspects whether a given construction c overlaps with any constructions in the set $\mathcal{C}$, returning a boolean value. We RANDOMKEEP($\cdot$) resolves conflicts by stochastically retaining either c or the conflicting construction in $\mathcal{C}$. ADD($\cdot$) appends non-overlapping constructions c to $\mathcal{C}$. This process is iteratively applied to all constructions in $\mathcal{C}$. The algorithm generates N sets of optimized constructions, $\mathcal{S} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N\}$, by applying the dynamic masking strategy N times. Finally, $\mathcal{S}$ captures diverse valid construction schemes.

Input and Output Definition. For a given grammatical sentence S_{c}, constructions are extracted to produce a masked sentence S_{m}:

$$S_{\mathrm{m}} = f_{\mathrm{c}}(S_{\mathrm{c}}, \mathcal{C}), \tag{1}$$

where $f_{\mathrm{c}}(\cdot)$ handles dynamic construction masking.

Training. The Seq2Seq model learns the mapping:

$$\hat{S}_{\mathrm{m}} = \mathrm{Seq2Seq}(S_{\mathrm{c}}), \tag{2}$$

optimizing the difference between $\hat{S}_{\mathrm{m}}$ and the target S_{m}.

Algorithm 1: Dynamic Masking for Multiple Construction Schemes

Input: The set of all constructions $\mathcal{C}$. Number of schemes N.
Output: A set of construction schemes $\mathcal{S} = \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_N\}$.

$\mathcal{S} \leftarrow \{\}$
for $i \in \{1, 2, \dots, N\}$ **do**
  $\mathcal{C}_i \leftarrow$ INITIALIZE()
  foreach *construction* $c \in \mathcal{C}$ **do**
    if CHECKOVERLAP(c, $\mathcal{C}_i$) **then**
      RANDOMKEEP(c, $\mathcal{C}_i$)
    end
    else
      ADD(c, $\mathcal{C}_i$)
    end
  end
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{\mathcal{C}_i\}$
end
return $\mathcal{S}$

Inference. During inference, the model inputs a sentence S, applies construction-based masking similarly, and outputs a CxG-masked sentence $\hat{S}_\mathrm{m}$ by aligning them with learned construction patterns:

$$\hat{S}_\mathrm{m} = \mathrm{Seq2Seq}(S) \tag{3}$$

3.2 CxGGEC Model

In this section, we present the training of CxGGEC models and the construction-guided GEC process. Our method includes three key steps: extending the vocabulary with constructions, preparing construction-masked parallel training data, and pretraining the model with the parallel data.

Construction Augmented Vocabulary. To integrate constructions into LMs, we explicitly extend their input vocabularies. Let $\mathcal{C}$ denote the set of all constructions extracted during preprocessing. Each construction $c_i \in \mathcal{C}$ is treated as a new token and added to the existing vocabulary $\mathcal{V}$. The updated vocabulary is denoted as $\mathcal{V}' = \mathcal{V} \cup \mathcal{C}$.

For the vocabulary extension, the embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$, where d is the embedding dimension, is updated to $\mathbf{E}' \in \mathbb{R}^{|\mathcal{V}'| \times d}$. All added construction embeddings are initialized randomly and fine-tuned during training. Specifically, for each construction c_i, its embedding is defined as:

$$\mathbf{e}_{c_i} = \mathrm{Initialize}(\mathrm{rand}(\mathbf{e}); \forall c_i \in \mathcal{C}), \tag{4}$$

where $\mathrm{rand}(\mathbf{e})$ generates random values sampled from a uniform distribution over $[-\sqrt{d}, \sqrt{d}]$.

Construction-Augmented Input Representation. To better leverage multiple construction predictions during training, we modify the input representation by concatenating the ungrammatical sentence $\mathbf{x}_{ug}$ with its masking-augmented sentences generated by construction prediction model.

Let $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$ denote the set of masked sentences generated by applying construction prediction model T times to $\mathbf{x}_{ug}$. The augmented input $\mathbf{x}'$ is then defined as:

$$\mathbf{x}' = \mathbf{x}_{ug} \oplus \mathbf{m}_1 \oplus \mathbf{m}_2 \oplus \cdots \oplus \mathbf{m}_T, \tag{5}$$

where $\oplus$ denotes sequence concatenation.

The inclusion of multiple masked sentences allows the model to benefit from diverse masking strategies and improves generalization.

The corresponding target sentence $\mathbf{y}$ is the standard grammatical correction for $\mathbf{x}_{ug}$. The parallel training pair is defined as $\langle \mathbf{x}', \mathbf{y} \rangle$, where $\mathbf{x}'$ is the construction-augmented input and $\mathbf{y}$ is the grammatical ground truth. This process generates a construction-augmented parallel corpus.

Pretraining with Construction-Augmented Examples. The pretraining phase uses the construction-augmented parallel corpus. The model's objective is to minimize the negative log-likelihood of the target sequence $\mathbf{y}$ conditioned on the input $\mathbf{x}'$. Formally, the loss function is defined as:

$$\mathcal{L} = -\sum_{i=1}^{N} \sum_{t=1}^{T} \log P(y_i^t \mid \mathbf{x}_i', y_i^{<t}; \Theta), \tag{6}$$

where y_i^t is the token at timestep t in the target sequence $\mathbf{y}_i$, T is the length of $\mathbf{y}_i$, and Θ are the model parameters. The probability $P(y_i^t \mid \cdot)$ is computed via the decoder's autoregressive output during training.

Pre-trained embeddings for vocabulary tokens remain initialized using the original model weights, while the embeddings for newly added construction tokens are learned adaptively.

4 Experiments

4.1 Experiments Setup

Datasets and Evaluation. For the English, we use the clean version of the original Lang-8 corpus (Mizumoto et al., 2011; Tajiri et al., 2012) as train sets. Specifically for the model based on Bart-Large model (Lewis et al., 2020), we use the

W&I+LOCNESS train-set (Bryant et al., 2019) for model fine-tuning following Zhang et al. (2022b). Following Zhang et al. (2022b), Li et al. (2023) and Li and Wang (2024), we use BEA-Dev (Bryant et al., 2019) as the development dataset, and use BEA-Test set and CoNLL14-Test set (Ng et al., 2014) as test datasets. For Chinese, following Li and Wang (2024), the models are fine-tuned on the Chinese Lang8 dataset (Zhao et al., 2018) and the HSK dataset (Zhang, 2009), and on the FCGEC training set (Xu et al., 2022) respectively. The models are evaluated on MuCGEC (Zhang et al., 2022a) and FCGEC test sets. For English evaluation, following Yuan et al. (2021a), we use ERRANT and M^2 (Dahlmeier and Ng, 2012) to evaluate GEC models on BEA-Test set and CoNLL14-Test set, respectively. For Chinese experiments, following Li and Wang (2024), models are evaluated on MuCGEC and FCGEC test sets using ChERRANT (Zhang et al., 2022a; Xu et al., 2022). Precision, recall, and $F_{0.5}$ values are reported metrics for all the experiments. Dataset details are listed in Appendix A.

Implementation. We train construction prediction model based on the BART-Base model(Lewis et al., 2020). For English GEC models, we train models based on the BART-Large (Lewis et al., 2020) and T5-Large (Raffel et al., 2020) models. Specifically, for the model based on the BART-Large, we refer to the training strategy of Zhang et al. (2022b). For the T5-Large model, we adopt the training strategy of Li et al. (2023). Both take Fairseq (Ott et al., 2019) as training framework. Due to the absence of a Chinese version of the T5 model, the experiments conducted in Chinese do not incorporate the T5 model. For creating Chinese construction inventory, we use Python library jieba (Feng, 2012) for sentence segmentation and part-of-speech tagging.

Baselines. (1) GECToR (Omelianchuk et al., 2020) represents the Seq2Edit models. (2) BART (Lewis et al., 2020) and T5 (Raffel et al., 2020) are backbones of Seq2Seq GEC methods. (3) SynGEC (Zhang et al., 2022b) incorporates syntactic information into the BART model. (4) Multi-Encoder (Yuan et al., 2021b) encodes error categories as auxiliary information. (5) GEC-DePend (Yakovlev et al., 2023) integrates error detection with correction by the MLM. (6) TemplateGEC (Li et al., 2023) uses the output of the GECToR model as supplementary information for Seq2Seq models. (7) DeCoGLM (Li and Wang, 2024) promotes performace of the GEC model by combining detection and correction tasks to mutually boost each other. The performance of GECToR and BART model on the Chinese dataset is reported by Li and Wang (2024), and the results for BART on the English dataset are reported by Zhang et al. (2022b).

4.2 Main Results

The main results of our experiments are listed in Table 2. It can be observed that our CxGGEC models achieve comparable performance across various benchmarks. Our framework demonstrates improvements across all benchmarks compared to the BART and T5 backbones. We achieve better performance than existing methods on the CoNLL14-Test set and FCGEC-Test set. The results show the effectiveness of our framework. Notably, our model based on the T5 backbone outperforms BART due to the basic idea of Raffel et al. (2020) to treat every text processing problem as a "text-to-text" problem, which can easily adapt to different inputs.

CxGGEC performs well on both English and Chinese GEC tasks, showcasing its generalizability in error correction across these two major languages. Compared to SynGEC, our method achieves further improvement on English datasets with less parameters added (13M), highlighting that constructions, as sets of slots, encode more semantic and syntactic information than only grammatical labels. This enables the model to achieve a deeper understanding of language usage and further enhances its GEC performance.

4.3 Analysis Study

Analysis on construction length. To explore the impact of construction length on the performance of GEC tasks, we apply two distinct methods to establish the construction inventory to support CxGGEC. First is the method of grammarinduction algorithm (Dunn, 2017), we refer to it as GIA for simplicity. The second method is CxGLearner (Xu et al., 2024).

The construction length distribution displayed in Figure 2 originates from the construction inventory covered in the CLang8-train dataset, a widely-used dataset for GEC models to align with the distribution patterns of sentences in English. The average construction length generated by GIA is approximately 3.0, while the constructions generated by CxGLearner exhibit a higher average length of 4.1. Notably, the lengths produced by CxGLearner ex-

Method	Parameters	English						Chinese					
		CoNLL-14 *test*			BEA-19 *test*			MuCGEC *test*			FCGEC *test*		
		P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$
GECToR	110M	**77.5**	40.1	65.3	**79.2**	53.9	72.4	46.72	27.14	40.83	46.11	34.35	43.16
BART-Large	400M	73.6	48.6	66.7	74.0	64.9	72.0	41.90	29.48	38.64	38.38	37.62	38.23
T5-Large	770M	-	-	66.1	-	-	72.1	-	-	-	-	-	-
SynGEC	110M+400M	74.7	49.0	67.6	75.1	65.5	72.9	**54.69**	29.10	**46.51**	-	-	-
Multi-Encoder	110M+107M	71.3	44.3	63.5	73.3	61.5	70.6	-	-	-	-	-	-
GEC-DePenD	253M	73.2	37.8	61.6	72.9	53.2	67.9	-	-	-	-	-	-
TemplateGEC	125M+770M	74.8	50.0	68.1	76.8	64.8	74.1	-	-	-	-	-	-
DeCoGLM	335M	75.1	49.4	68.0	77.4	64.6	**74.4**	45.01	**31.77**	41.55	55.75	37.91	50.96
CxGGEC (Bart-large)	13M+400M	73.8	50.5	67.6	74.8	65.3	72.7	47.90	29.94	42.78	**59.90**	35.92	**52.84**
CxGGEC (T5-large)	13M+770M	74.9	**50.7**	**68.3**	75.7	**65.8**	73.5	-	-	-	-	-	-

Table 2: Results on English and Chinese GEC benchmarks. The highest metric is indicated in bold.

Strategy	BEA-19			CoNLL-14		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
GIA	74.0	65.2	72.1	73.7	50.2	67.4
CxGLearner	**75.7**	**65.8**	**73.5**	**74.9**	**50.7**	**68.3**

Table 3: Performance of CxGGEC (T5-large) with different construction inventory establishing strategies on BEA-19 test and CoNLL-14 test benchmarks.

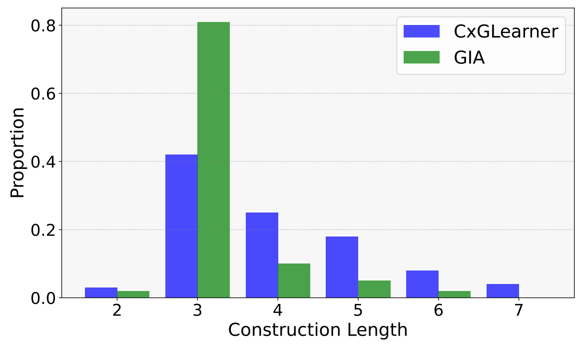

Figure 2: Length distribution of construction inventories extracted from GIA (Dunn, 2017) and CxGLearner (Xu et al., 2024) .

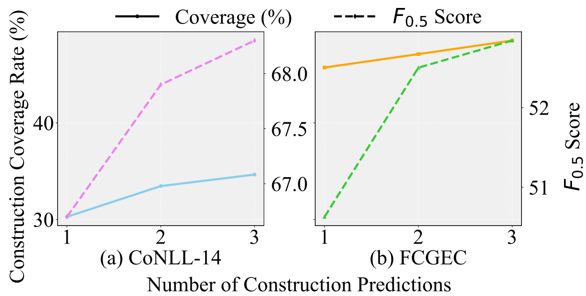

Figure 3: Construction coverage rate and $F_{0.5}$ score across prediction steps.

hibit a more balanced distribution. As shown in Table 3, the constructions generated by CxGLearner provide more significant guidance for the LM GEC task compared to GIA.

This observation implies that CxGLearner achieves more comprehensive coverage of constructions inherent in corpus. While both methods generate useful constructions for GEC, constructions extracted with GIA tend to be relatively short or incomplete, because GIA is prone to truncate the constructions too early. This result indicates that balanced length distribution and semantic completeness of constructions lead to better performance on GEC tasks, because they align with the usage patterns in the corpus and contain more knowledge of language usage.

Analysis on Construction Coverage. To reveal how construction coverage contributes to GEC tasks, we perform experiments on number of construction predictions in Figure 3. We observe a gradual improvement in GEC performance as the number of predictions increases. The construction coverage rate is defined as the ratio of the number of sentences of which the constructions identified cover the error positions to the total number of sentences. The result shows that increasing construction predictions enhances the model's ability to cover sentence errors effectively, and therefore improve the overall performance of GEC tasks.

Analysis on Construction Masking Strategy. To figure out the impact of dynamic masking strategy on GEC tasks, we analyze the results of the GEC task without dynamic masking strategy compared to results of CxGGEC in Table 4. We refer to dynamic masking as DM for simplicity. The results demonstrate that DM yields superior model performance compared to fixed masking. This can be attributed to the ability of DM to prevent construc-

Strategy	BEA-19			CoNLL-14		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
CxGGEC	**75.7**	65.8	**73.5**	**74.9**	**50.7**	**68.3**
w/o DM	74.1	**66.9**	72.5	73.5	49.8	67.1

Table 4: Comparison of the performance of CxGGEC (T5-large) with and without dynamic masking.

Type	Baseline			CxGGEC		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
M	72.4	65.8	71.0	74.2	70.0	**73.3**
R	72.0	60.6	69.4	73.4	63.0	**71.1**
U	75.0	69.5	73.8	75.2	70.6	**74.2**

Table 5: Results of error types in BEA-Test. Baseline is the T5-Large model. (M, R, and U stand for missing, replacement, and unnecessary errors, respectively.)

tion prediction model from overfitting to specific masking patterns and to enhance the model's capacity to adapt to diverse contexts.

Analysis on Error Types. To reveal what types of error can CxG guide GEC tasks better, we compare results of error types on the BEA-Test benchmark in Figure 5. The baseline is T5-large model and the CxGGEC model is based on T5-Large model. Overall, CxGGEC demonstrates higher performance on three error types, particularly in missing and replacement errors but achieves subtle improvement in unnecessary errors. The potential reason is that constructions identified by the prediction model may fail to include unnecessary errors. This requires the model to expend effort on error detection and correction, thereby resulting in only subtle improvement.

Analysis on POS Tags. We intend to explore the impact of part-of-speech (POS) tags on the BEA-Test dataset. UPOS stands for Universal POS tags and XPOS stands for Language-Specific POS tags. We compare the results of using only UPOS, using only XPOS, and combining the two with a specified proportion during training construction prediction model to evaluate their effectiveness . As shown in Table 6, using only UPOS performs slightly worse than using only XPOS, because XPOS is better at capturing fine-grained grammatical and structural information. The combination of UPOS and XPOS yields better results because adding a certain proportion of UPOS provides high-level abstraction that aids in capturing generalized linguistic patterns. This combination enables the model to balance generalization and specificity, ultimately enhancing its overall performance.

UPOS	XPOS	BEA-19			CoNLL-14		
		P	R	$F_{0.5}$	P	R	$F_{0.5}$
✗	✗	69.2	48.4	66.5	71.1	47.7	65.1
✓	✗	71.4	63.2	69.6	73.3	47.4	66.1
✗	✓	72.8	64.4	70.9	74.5	48.7	67.4
✓	✓	**75.7**	**65.8**	**73.5**	**74.9**	**50.7**	**68.3**

Table 6: Results of POS tags.

Analysis on Visualization. To explain why CxG can effectively guide GEC tasks from the perspective of language models, we compare the attention matrices of a baseline LM (Bart-Large) and CxGGEC model based on Bart-Large model in Figure 4. Tokens identified as constructions (construction-masked segments) are highlighted in red, while the shaded area further emphasizes the attention on these tokens. The result shows that attention of CxGGEC model focuses around phrases, especially those involving constructions (highlighted parts). This reflects the ability of the CxGGEC model to incorporate constructional information from constructions, guiding the model to focus on meaningful sections of the sentence rather than isolated tokens. This allows CxGGEC to better interpret the overall context, particularly in ungrammatical sentences, where individual tokens may not provide sufficient information.

5 Related Works

GEC Methods. Two widely used approaches in GEC are Seq2Edit and Seq2Seq. In Seq2Edit methods, Seq2Edits (Stahlberg and Kumar, 2020) predicts a sequence of span-level edit operations applied to the source text, while GECToR (Omelianchuk et al., 2020) extends traditional operations with custom transformations, such as suffix changes and token merging. The advantage of the Seq2Edit approach is its faster speed compared to Seq2Seq. However, a key limitation is its reliance on manually curated editing operations, which can reduce transferability and fluency (Li et al., 2022). Seq2Seq models (Lewis et al., 2020; Raffel et al., 2020) have demonstrated high performance in GEC (Junczys-Dowmunt et al., 2018b; Choe et al., 2019; Zhao et al., 2019; Katsumata and Komachi, 2020), though their inference efficiency is lower compared to Seq2Edit. Mallinson et al. (2020) and Yakovlev et al. (2023) utilize Masked Language Models (Kenton and Toutanova, 2019)

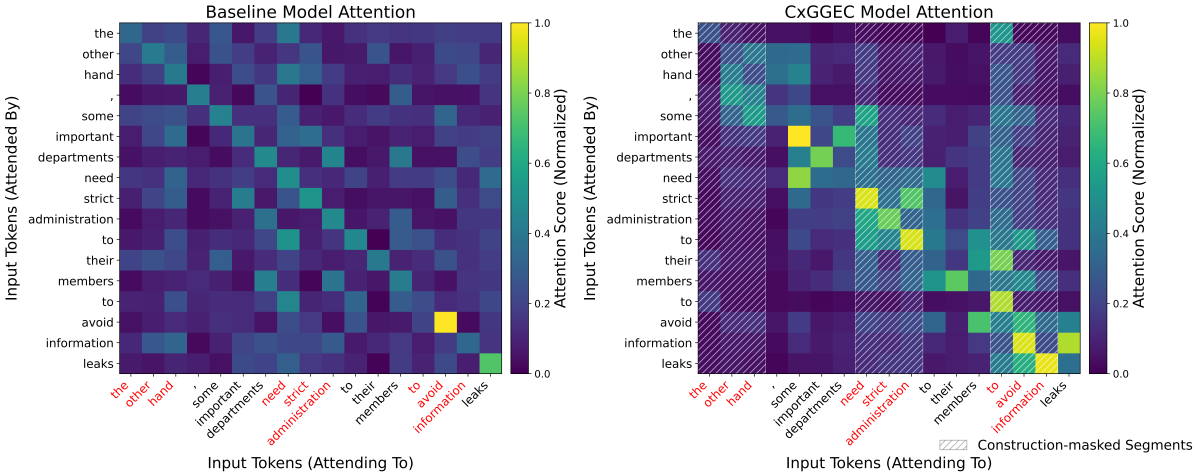

Figure 4: Comparison of attention maps based on Bart-Large and CxGGEC (Bart-Large).

to generate corrections, aiming to benefit from self-supervised pretraining. Previous studies have also incorporated error detection results (e.g., detection labels from a Seq2Edit model) as auxiliary information to enhance GEC performance (Kaneko et al., 2020; Yuan et al., 2021b; Li et al., 2023). State-of-the-art models further incorporate syntactic information to improve performance. For example, SynGEC (Zhang et al., 2022b) integrates dependency syntax into GEC models, while CSynGEC (Zhang and Li, 2022) enhances GEC tasks by leveraging constituent-based syntax. However, current methods rely on grammatical labels for syntactic information, failing to fully capture the structural and semantic usage patterns of a language. Therefore, we introduce construction grammar to address the issue.

Applications of CxG in NLP. Construction Grammar (CxG) has been explored in natural language processing tasks. Kiselev (2020) constructs a CxG-based knowledge network for a deeper understanding of text. Dunn (2023) employs constructions to model variation across and dialects. Xu et al. (2023) leverage constructional information to enrich language representation for natural language understanding tasks. Subsequently, Xu et al. (2024) encode constructions as inductive biases to explicitly embed constructional semantics and guide language modeling. However, there has been no effort to ascertain whether constructions can provide benefits in guiding GEC tasks. Our work aims to bridge this gap.

Construction Inventory Establishment. An inventory of constructions serves as a valuable resource for CxG-based research. Several construction inventories have been created for various languages (e.g., English, German) by lexicographers and linguists (Lyngfelt et al., 2018), primarily through manual development, which is labor-intensive and depends on expert experience. Weissweiler et al. (2024) utilize GPT-3.5 and propose a hybrid human-LLM corpus construction method, with a focus on the caused-motion construction. To establish a comprehensive construction inventory automatically from corpora, Dunn (2017) proposes a grammar induction algorithm based on the computation of associations between adjacent words using a hard threshold. To generate more complete constructions, Xu et al. (2024) introduce a LM-based approach to assess slot constraints over longer distances. However, these methods are unable to extract potential constructions from ungrammatical sentences. To this end, we propose a construction prediction model designed to identify expected constructions directly from ungrammatical sentences.

6 Conclusion

In this paper, we propose a construction-guided grammatical error correction approach (CxGGEC) that leverages construction grammar (CxG) to enhance error detection and correction. Our framework involves three key steps: (1) generating a comprehensive construction inventory using CxGLearner, (2) identifying constructions in ungrammatical sentences through a noise-tolerant language model, and (3) guiding the GEC task by integrating construction-masked sentences into the training process. Extensive experiments on both English and Chinese GEC benchmarks demonstrate the effectiveness of CxGGEC.

Limitations

In this study, the limitations can be summarized into two major aspects:

(1) Increased input length and slower inference speed. Incorporating constructional information into the model input increases the overall input length, which inevitably slows down the inference speed. This trade-off between additional linguistic information and computational efficiency poses a challenge, especially for real-time or large-scale applications.

(2) Randomness in construction prediction. The construction-prediction model exhibits a degree of randomness. Even though the use of dynamic masking strategies improves the model's ability to generate diverse constructions, it cannot guarantee that the generated constructions fully cover all errors in every prediction. To address this limitation, multiple rounds of inference could be applied to enhance construction coverage for uncovered errors, potentially further improving GEC performance.

Ethics Statement

In this work, we use publicly available corpora and benchmarks under their licenses. These publicly available data are checked to ensure that they do not include any offensive and illegal content.

Acknowledgements

We thank all anonymous reviewers for their insightful comments and suggestions. And we also appreciate CxGLearner for providing us with the automatic extraction method of constructions. 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 *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4260–4270, Hong Kong, China. Association for Computational Linguistics.

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. 2023. Grammatical error correction: A survey of the state of the art. *Computational Linguistics*, 49(3):643–701.

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

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572, Montréal, Canada. Association for Computational Linguistics.

Jonathan Dunn. 2017. Computational learning of construction grammars. *Language and cognition*, 9(2):254–292.

Jonathan Dunn. 2019. Frequency vs. association for constraint selection in usage-based construction grammar. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 117–128, Minneapolis, Minnesota. Association for Computational Linguistics.

Jonathan Dunn. 2023. Exploring the constructicon: Linguistic analysis of a computational CxG. In *Proceedings of the First International Workshop on Construction Grammars and NLP (CxGs+NLP, GURT/SyntaxFest 2023)*, pages 1–11, Washington, D.C. Association for Computational Linguistics.

Junyi Feng. 2012. Jieba: Chinese text segmentation. GitHub repository. Accessed: October 2023.

Adele E Goldberg. 1995. Constructions: A construction grammar approach to argument structure. *University of Chicago*.

Adele E Goldberg. 2003. Constructions: A new theoretical approach to language. *Trends in cognitive sciences*, 7(5):219–224.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Ting Jiang, Deqing Wang, Leilei Sun, Huayi Yang, Zhengyang Zhao, and Fuzhen Zhuang. 2021. Lightxml: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7987–7994.

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

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

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254, Online. Association for Computational Linguistics.

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

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1. Minneapolis, Minnesota.

Denis Kiselev. 2020. An ai using construction grammar: Automatic acquisition of knowledge about words. In *ICAART (2)*, pages 289–296.

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

Jiquan Li, Junliang Guo, Yongxin Zhu, Xin Sheng, Deqiang Jiang, Bo Ren, and Linli Xu. 2022. Sequence-to-action: Grammatical error correction with action guided sequence generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10974–10982.

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

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.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Benjamin Lyngfelt, Lars Borin, Kyoko Ohara, and Tiago Timponi Torrent. 2018. *Constructicography: Constructicon development across languages*, volume 22. John Benjamins Publishing Company.

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

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

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, Baltimore, Maryland. Association for Computational Linguistics.

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

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

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

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5937–5947, Online. Association for Computational Linguistics.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Zhaohong Wan and Xiaojun Wan. 2021. A syntax-guided grammatical error correction model with dependency tree correction. *arXiv preprint arXiv:2111.03294*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Leonie Weissweiler, Abdullatif Köksal, and Hinrich Schütze. 2024. Hybrid human-llm corpus construction and llm evaluation for rare linguistic phenomena. *Preprint*, arXiv:2403.06965.

Lvxiaowei Xu, Zhilin Gong, Jianhua Dai, Tianxiang Wang, Ming Cai, and Jiawei Peng. 2024. Coelm: Construction-enhanced language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10061–10081.

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

Lvxiaowei Xu, Jianwang Wu, Jiawei Peng, Zhilin Gong, Ming Cai, and Tianxiang Wang. 2023. Enhancing language representation with constructional information for natural language understanding. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 4685–4705. Association for Computational Linguistics.

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

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

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

Baolin Zhang. 2009. Features and functions of the hsk dynamic composition corpus. *International Chinese Language Education*, 4:71–79.

Yue Zhang and Zhenghua Li. 2022. Csyngec: Incorporating constituent-based syntax for grammatical error correction with a tailored gec-oriented parser. *Preprint*, arXiv:2211.08158.

Yue Zhang, Zhenghua Li, Zuyi Bao, Jiacheng Li, Bo Zhang, Chen Li, Fei Huang, and Min Zhang. 2022a. MuCGEC: a multi-reference multi-source evaluation dataset for Chinese grammatical error correction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3118–3130, Seattle, United States. Association for Computational Linguistics.

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

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 156–165, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Algorithm 2: Fixed Masking Using Maximum Coverage

Input: A set of construction schemes $\mathcal{S} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N\}$. Sentence $\mathcal{S}_{sent}$.
Output: The optimal set $\mathcal{C}_O$.

```
C_O ← {}
maxCoverage ← 0
foreach scheme C_i ∈ S do
    coverage ← CALCULATECOVERAGE(C_i, S_sent)
    if coverage > maxCoverage then
        maxCoverage ← coverage
        C_O ← C_i
    end
end
return C_O
```

A Datasets Used in GEC Models

Dataset	#Sentences	%Error	Usage
CLang8	2,372,119	57.8	Pre-training (†, ‡)
W&I+LOCNESS	34,308	66.3	Fine-tuning (†)
BEA19-Dev	4,384	65.2	Validation (†,‡)
CoNLL14-Test	1,312	72.3	Testing (†,‡)
BEA19-Test	4,477	-	Testing (†,‡)

Table 7: Statistics of English GEC datasets. #Sentences denotes the number of sentences.%Error refers to the proportion of erroneous sentences. †: indicates usage for model based on BART-Large model. ‡: indicates usage for model based on T5-Large model.

Dataset	#Sentences	%Error	Usage
Lang8	1,220,906	89.5	Training
HSK	15,687	60.8	Training
FCGEC-train	36,340	54.5	Training
MuCGEC-dev	1,125	95.1	Validation
MuCGEC-test	5,938	92.2	Testing
FCGEC-test	3,000	54.5	Testing

Table 8: Statistics of Chinese GEC datasets.

B Training Data Examples

We use construction-masked sentences concatenated with the original ungrammatical sentences as inputs to the GEC model and pair them with ground-truth sentences to form parallel corpora for GEC model training. Examples are shown in Table 9.

C Fixed Masking Strategy

Compared to dynamic masking to the train construction prediction model, fixed masking we use can be demonstrated in Algorithm 2. The algorithm examines a predefined set of construction schemes and selects the one that maximizes the area of constructions within the given sentence. The input to the algorithm consists of a set of construction schemes $\mathcal{S} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N\}$ and a sentence $\mathcal{S}_{sent}$. The algorithm iteratively evaluates each construction scheme $\mathcal{C}_i \in \mathcal{S}$ to calculate its coverage over the input sentence, relying on the function CALCULATECOVERAGE. The goal is to identify the construction scheme $\mathcal{C}_O$ that achieves the highest coverage with respect to the constructions inherent in the sentence. The 'maxCoverage' value is updated whenever a scheme $\mathcal{C}_i$ with higher coverage is encountered, and $\mathcal{C}_O$ is set to $\mathcal{C}_i$. Finally, the algorithm returns $\mathcal{C}_O$, which represents the optimal construction masking scheme. However, fixed masking is not conducive to improving the construction prediction model's generalization performance. Therefore, in comparison, dynamic masking was chosen as a better alternative according to results in Table 4.

D Construction Inventory Establishment

In this task, we utilize CxGLearner to build the construction inventory. CxGLearner is an unsupervised system that can autonomously extract high-quality constructions from text corpora. It has three main parts working in a sequence. To begin with, the preprocessor and multi-level encoder clean the raw text and break it into tokens, creating slots with abstract representations at different levels (lexical, UPOS and XPOS) based on the tokens. Then, the association strength estimator, which is based on GPT-2 and pre-trained on a large-scale corpus, evaluates how strongly the candidate sequences are associated with the slots. Following that, a candidate construction extractor and pruner are used to arrange slot sequences that meet the associative requirements into a set of potential construction candidates and remove redundant and irregular constructions.

To demonstrate clearly, let $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ denote a raw text corpus, where each d_i is a document. CxGLearner extracts a set of constructions $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$ through an unsupervised pipeline consisting of the following components:

Example	Input Sentence (Original Sentence + Construction-Masked Sentence)	Ground-Truth Sentence
Example 1	About winter [SEP] <ADP>–<NN>–<NOUN>	About winter
Example 2	This is my second post . [SEP] This <VBZ>–<PRON>–<ADJ> post .	This is my second post .
Example 3	People usually get this kind of hypertesion after they become adult . [SEP] People usually get this kind of hypertesion <IN>–Ġthey–<VBP> adult .	People usually get this kind of hypertesion when they become adult .
Example 4	After the initial ceremony , the group photo was taken . [SEP] After <DT>–<JJ>–<NOUN> , <DET>–<NN>–<NOUN> was taken .	After the initial ceremony , the group photo was taken .
Example 5	One time , I had an Japanese examination . [SEP] One time , I had <DT>–<JJ>–<NOUN> .	One time , I had a Japanese examination .

Table 9: Examples of construction-masked sentences paired with ground-truth sentences for GEC training.

Preprocessing and Multi-level Encoding Each document d_i is tokenized into a sequence of tokens $\mathbf{t}_i = (t_1, \ldots, t_{M_i})$. For each token t_j, we generate abstract slot representations at three levels:

$$\mathbf{e}_j^{\text{lex}} = \text{Embed}(t_j), \tag{7}$$

$$\mathbf{e}_j^{\text{upos}} = \text{UPOS}(t_j), \tag{8}$$

$$\mathbf{e}_j^{\text{xpos}} = \text{XPOS}(t_j). \tag{9}$$

The encoded sequence is $\mathbf{E}_i = (\mathbf{e}_1, \ldots, \mathbf{e}_{M_i})$, where $\mathbf{e}_j = [\mathbf{e}_j^{\text{lex}}; \mathbf{e}_j^{\text{upos}}; \mathbf{e}_j^{\text{xpos}}]$.

Association Strength Estimation A pretrained GPT-2 model computes the association score for a candidate slot sequence $\mathbf{s} = (s_1, \ldots, s_L)$ as:

$$\text{Assoc}(\mathbf{s}) = \frac{1}{L} \sum_{k=1}^{L-1} \log P(s_{k+1} \mid s_{\leq k}; \theta), \tag{10}$$

where $s_{\leq k} \equiv s_1, \ldots, s_k$, and θ denotes GPT-2 parameters.

Candidate Extraction and Pruning Let $\mathcal{S}_{\text{cand}}$ be candidates with $\text{Assoc}(\mathbf{s}) \geq \tau$. The pruned set is:

$$\mathcal{S}_{\text{pruned}} = \big\{ \mathbf{s} \in \mathcal{S}_{\text{cand}} \;\big|\; \text{Sim}(\mathbf{s}, \mathbf{s}') < \epsilon, \; \forall \mathbf{s}' \neq \mathbf{s} \big\}, \tag{11}$$

where $\text{Sim}(\cdot)$ measures structural similarity.