System Report for CCL23-Eval Task 8: Chinese Grammar Error Detection and Correction Using Multi-Granularity Information

Yixuan Wang, Yijun Liu, Bo Sun, Wanxiang Che[*]
Research Center for Social Computing and Information Retrieval (SCIR),
Harbin Institute of Technology, China
{yixuanwang, bsun, car}@ir.hit.edu.cn
7203610630@stu.hit.edu.cn

Abstract

This paper introduces our system at CCL-2023 Task: Chinese Essay Fluency Evaluation (CEFE). The CEFE task aims to study the identification and correction of grammatical errors in primary and middle school students' test compositions. The evaluation has three tracks to examine the recognition of wrong sentence types, character-level error correction, and wrong sentence rewriting. According to the task characteristics and data distribution of each track, we propose a token-level discriminative model based on sequence labeling for the multi-label classification task of wrong sentences, an auto-encoder model based on edited labels for character-level error correction and a seq2seq model obtained by pre-training on pseudo data and fine-tuning on labeled data to solve the wrong sentence rewriting task. In the final evaluation results, the method we proposed won the first place in all three tracks according to the corresponding evaluation metrics.

1 Introduction

With the development of the Internet, the scale of online texts is also increasing, and it is difficult to meet the needs of text proofreading by relying on manual review alone. Especially in some error-intensive fields, such as the composition evaluation of elementary and middle school students, manual evaluation has become expensive and inefficient. At this time, it becomes very necessary to use deep learning technology to build an efficient evaluation system, which can assist teachers in identifying.

In order to promote the development of the field of text error correction, the China National Conference on Computational Linguistics (CCL-2023) has taken Chinese Essay Fluency Evaluation (CEFE) as one of the shared tasks. This task systematically classifies text errors at different granularities, provides human-annotated data, and proposes three tracks covering error correction and error detection.

In this work, we introduce our method at CCL-2023 CEFE task. For error detection, we adopt a fine-grained error detection model based on sequence annotation. Sentence-level multi-label tasks are accomplished by discriminating the type of error involved in each token. At the same time, we use techniques such as model inheritance and threshold post-processing to alleviate the bias caused by pseudo-data training. Due to the misalignment between the sequence labeling task and the provided human-annotated data, we constructed a large amount of pseudo-data for various types of errors based on LTP(Che et al., 2020) and heuristic rules, which were used for the training of the Track1 model and the pre-training of Track2 and Track3 models. For error correction, we trained an auto-encoder model based on edit label prediction and an auto-regressive model of seq2seq for character-level errors and extensive errors (including character-level and component-level), respectively. In the final evaluation, our method won the first place in the three tracks of wrong sentence type discrimination, character-level error correction, and wrong sentence rewriting.

This article is organized as follows: Section 2 briefly introduces the CEFE shared task; Section 3 mainly expounds the methods we use in this evaluation, including data level and model level; Section 4

[*]Email corresponding.

presents the main experimental results; Section 5 introduces some related work on text error correction; Finally, we conclude in Section 6 with reflections on future work.

2 Chinese Essay Fluency Evaluation

The goal of the CCL2023 CEFE task is to identify wrong sentences in primary and middle school students' compositions, judge the wrong sentence category they belong to, and propose amendments.

The previous work did not carry out a detailed classification of wrong sentences. This evaluation is the first to start from the two perspectives of character-level errors and component-level errors, and defines four categories of coarse-grained error categories (Character-level error, Component Incompleteness error, Component Redundancy error, and Component Mismatch error) , which contain 14 fine-grained error categories, as shown in Table 1.

Coarse-grained Error	Fine-grained Error	Amount of Pseudo Data
Character-level	Missing Word	20w
	Typo	27w
	Missing Punctuation	16w
	Punctuation Misuse	20w
Component Incompleteness	Subject unknown	5w
	Predicate Incompleteness	5w
	Object Incompleteness	5w
	Other Incompleteness	5w
Component Redundancy	Subject Redundancy	1k
	Function Word Redundancy	1k
	Other Redundancy	1k
Component Mismatch	Improper Word Order	1k
	Verb-object Mismatch	1k
	Other Mismatch	1k

Table 1: The 4 types of coarse-grained error categories and their corresponding 14 fine-grained error categories provided in this evaluation, among which Track 1 and Track 3 involve all categories, and Track 2 only involves Character-level error category. The table also shows the number of pseudo data we constructed for each error category, which will be explained in 4.2.

Specifically, Track 1 of the task is mainly dedicated to identifying the error types of wrong sentences, Track 2 requires the identification and correction of Character-level coarse-grained errors in sentences, and Track 3 requires the rewriting of wrong sentences containing extensive errors.

3 Methodology

According to the requirements of each track, the system we submitted needs to complete the identification and error correction of wrong sentences. Below we will introduce our method from these two aspects.

3.1 Wrong Sentence Type Recognition

Track 1 is a sentence-level multi-label classification task, which requires the model to determine the coarse-grained error and fine-grained error categories contained in the wrong sentence. However, due to the large number of types of wrong sentences and the large differences in the scale of various types of errors (a token is involved at the character level, and a span is usually involved at the component level), using conventional multi-label classification methods has not achieved good results.

Therefore, we consider classifying wrong sentences from the token level rather than the sentence level. The token-level sequence labeling task can not only convert sentence-level multi-label classification into token-level classification tasks (different errors usually involve different characters), but also simplify relatively difficult tasks (such as Component Mismatch) through interpretable token labeling. Specifically, the sequence labels for Character-level, Component Incompleteness, Component Redundancy, and Component Mismatch are shown in Figure 1.

Coarse-grained Category	Sequence Label
Character-level	感受秦 [谁 #char_error] 河所承载的历史
Component Incompleteness	试验了6000 [多 #miss_other] 材料
Component Redundancy	[我 #redu_sub] 过了一会，我完成了测试
Component Mismatch	但也 [达到 #coll_vobj] 了很大 [进步 #coll_vobj]

Figure 1: Illustration of the sequence labels for each category. The correct character is marked as **#correct**, and all **#correct** labels are omitted in the table for clarity.

Sequence Labeling Method

According to the error category definition of Track1, our sequence labeling model has a total of 15 categories (correct label and other fine-grained error labels). Through the sequence labeling model, we obtain fine-grained error labels for each token. Finally, we integrate all the involved fine-grained labels as the result of the sentence-level fine-grained category, and then deduce the coarse-grained category according to the fine-grained category. The entire pipeline processing flow is shown in Figure 2.

Pseudo Data Construction

Since the human-annotated data provided by Track1 and the sequence labeling task are not aligned, and the amount of each track's data is not large. We consider using some semantic parsing tools and open-source data to construct a large amount of pseudo data for model pre-training.

Finally, we use LTP(Che et al., 2020) and some heuristic rules to construct a parallel corpus containing corresponding errors for the correct sentences in the CGED(Rao et al., 2018) training set. The specific rules are as follows:

1. **Character-level** error's construction mainly includes additions, deletions, and modifications to the original text. We construct **Missing Word** and **Missing Punctuation** by random delete operations. Relying on the word confusion sets proposed by Wang et al. (2018) and the Pinyin confusion set collected from the Internet, we construct **Typo** and **Punctuation Misuse** errors. We also randomly inserted some words from the vocabulary to cover the case of redundant word errors, whether it is Chinese characters or punctuation.

2. **Component Incompleteness** error's construction mainly depends on the syntactic analysis of LTP. According to the syntactic analysis results of LTP, we randomly delete the subject, object, predicate, and other component in it to construct **Subject unknown**, **Predicate Incompleteness**, **Object Incompleteness**, and **Other Incompleteness**. It should be noted that **Subject Unknown** also contains error subject sentences, so we will also randomly replace some subjects to construct this type of pseudo data.

3. **Component Redundancy** error's construction also depends on the syntactic analysis of LTP. We construct **Subject Redundancy** and **Function Word Redundancy** by repeating or inserting the corresponding components. **Other Redundancy** is difficult to construct, so we directly use the corresponding part of the open-source Chinese semantic error dataset (CSED) proposed by Sun et al. (2023) as the training set.

4. **Component Mismatch** error mainly includes three types of errors. **Improper Word Order** error can be achieved by randomly shuffling spans, and we also use some of this type of data in CSED. For **Verb-object Mismatch** and **Other Mismatch** errors, we first constructed some subject-verb, verb-object collocation knowledge bases according to the LTP analysis results, and then we randomly replaced the subject, predicate, object in sentences from the knowledge base to realize improper collocation errors.

Transformer Model for Sequence Labeling

We use the model of the Transformer architecture (Vaswani et al., 2017) to complete the sequence labeling task. As usual, we first use the encoder model to model the input token to obtain hidden layer features, and then, we predict the label type of each token through the linear layer. For the input sequence: $S = t_0, t_1, ..., t_n$, the formula for the labeling process is as follows:

$$h_i^0 = W_e t_i + W_p \tag{1}$$

$$h_i^l = TransformerBlock(h_i^{l-1}) \tag{2}$$

$$y_i = Softmax(W_{linear} h_i^l + b) \tag{3}$$

where t_i is the current token, W_e is the word embedding matrix, and W_e is the position embedding matrix. After the extraction of each block, the hidden layer representation of the L layer h_i^l is obtained to predict the final label y_i by linear layer W_{linear}.

During the training stage, we use the label cross-entropy loss to optimize the model, which can be formulated as follow:

$$loss_{sequence\_labeling} = \sum_{i=1}^{n} CrossEntropy(y_i, \hat{y_i}) \tag{4}$$

where y_i represents the model prediction result, and $\hat{y_i}$ represents the gold label.

Model Integration

Due to the large difference in the scale of the four coarse-grained category errors, we found that a single model can't discriminate all errors well. Therefore, we consider using four different sequence labeling models to model errors of different scales. The specific method we use for inheriting model results can be formulated as follows:

$$Final_i = \bigcup_{m=1}^{4} Pred_m \tag{5}$$

where $Pred_m$ is the set of fine-grained error categories predicted by the mth model. For efficiency, we directly union the results of each model at the sentence level. This enables our system to learn more specifically about different types of errors and make more reasonable judgments.

Threshold Filtering

Although through model ensembles, we have been able to guarantee the requirement on recall. Our model still suffers from excessive false positives, due to the bias of pseudo data and the difficulty of the task.

Therefore, we consider the post-processing operation of threshold filtering on the model output.

We set a bound of confidence. When the predicted label is not correct and its corresponding confidence value is less than this lower limit, set this label to correct.

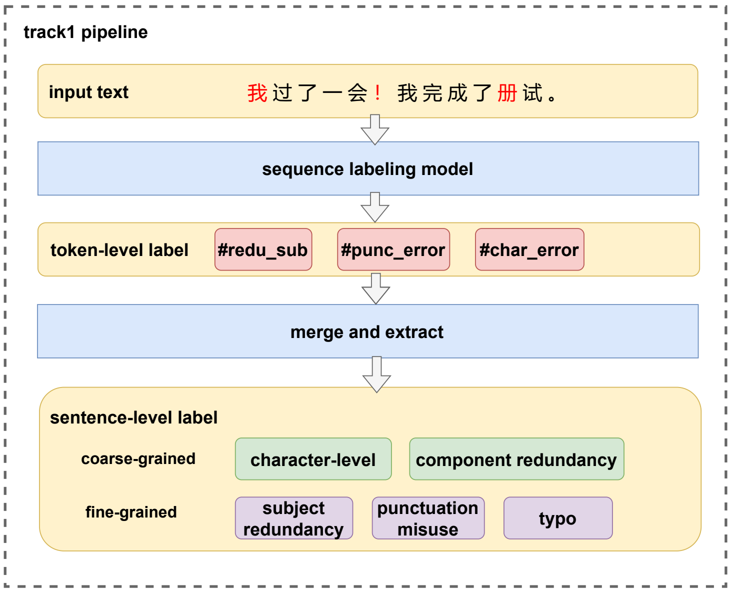

Figure 2: Illustration of the pipeline for the detection of the entire track1 wrong sentence category. First, we obtain the token-level error labels involved in the input text through the sequence labeling model, and then obtain the sentence-level results by merging and extracting the token-level labels.

Specifically, we set a confidence hyperparameter as a threshold. When the model confidence is lower than this hyperparameter, we will set it as correct. The filter formula is as follows:

$$label_{ij} = \begin{cases} y_{ij}, & \text{if } P(y_{ij}) > threshold \\ correct, & \text{else} \end{cases} \tag{6}$$

where y_{ij} is the predicted label, and $P(y_{ij})$ is the model's confidence in the label.

3.2 Wrong Sentence Correction

In addition to detecting the wrong sentence category, Track2 and Track3 also require us to correct the sentences to varying degrees. Considering the difference between these two categories of errors (the edit distance involved in Character-level errors is relatively short, usually within 1 word, while the edit distance involved in component-level errors is usually longer), We use two model architectures, auto-encoder and auto-regressive, to accomplish these two error correction tasks, respectively.

Operation	Detail	explanation
insert	repeat	insert a repeated word or token (including punctuation)
	redundancy	insert a synonym, based on bigcilin[0]
	random	insert a random word from the vocabulary
delete	random	delete a token (including punctuation)
replace	token-level	replace based on token-level confusion set (Wang et al., 2018)
	word-level	replace based on word-level confusion set[1]
	random	randomly replace from the vocabulary

Table 2: Introduction to the method of constructing Character-level error pseudo data

[0]www.bigcilin.com
[1]constructed from token-level confusion set

Pseudo Data Construction

As in the error detection part, in view of the relatively small number of training sets provided by the evaluation, we used a similar LTP-based method (see in 3.1) to construct some pseudo data for the alignment error correction task in the pre-training stage.

In particular, we have constructed more fine-grained pseudo data for Character-level errors. The main construction methods are shown in Table 2. We randomly process the unlabeled corpus according to the operations in the table, so as to obtain sentences containing corresponding errors.

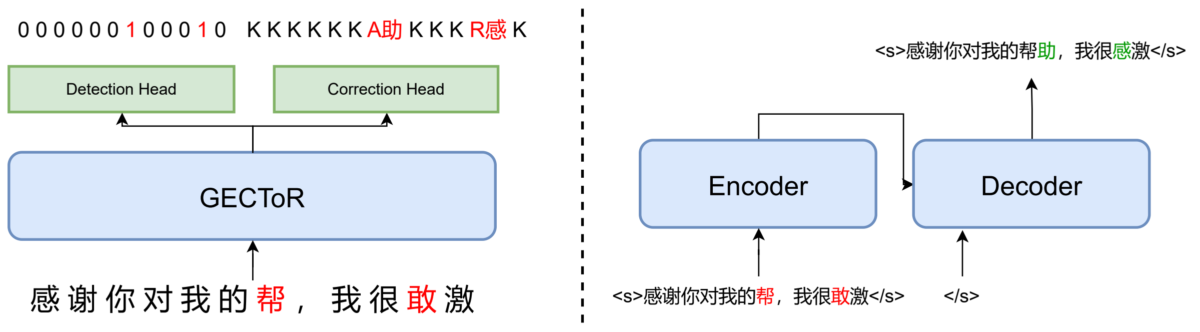
0 0 0 0 0 0 1 0 0 0 1 0
K K K K K K A助 K K K R感 K
Detection Head
Correction Head
GECToR
感 谢 你 对 我 的 帮 ， 我 很 敢 激
<s>感谢你对我的帮助，我很感激</s>
Encoder
Decoder
<s>感谢你对我的帮，我很敢激</s>
</s>

Figure 3: Illustration of the multi-task GECToR model used in Track2 and the seq2seq model used in Track3. For error correction header prediction labels, **A** means insertion and **R** means replacement.

Character-level Error Correction

For Character-level errors, we use the GECToR framework (Omelianchuk et al., 2020) based on editing tag sequence annotations to implement addition, deletion, and modification operations for Chinese characters and punctuation errors. We believe that non-generative models are easier to fit to the training set than generative models, and can make more conservative modifications.

In order to obtain a reliable Chinese GECToR model, we initialized with the weights of Chinese Bert(Cui et al., 2021), pre-trained on pseudo data and fine-tuned on the provided real distribution training set. We use error detection and correction multi-task learning to train the model. The model architecture is shown in the left side of Figure 3. As shown in the figure, we use the loss of multi-task learning as the optimization target during our GECToR model's training stage, which can be formulated as follows:

$$loss_{detect} = \sum_{i=1}^{n} CrossEntropy(y_{bi}, \hat{y}_{bi}) \tag{7}$$

$$loss_{correct} = \sum_{i=1}^{n} CrossEntropy(y_{token}, \hat{y}_{token}) \tag{8}$$

$$loss_{gector} = loss_{detect} + loss_{correct} \tag{9}$$

where $y_{bi}, \hat{y}_{bi}$ represents the binary label of the model prediction and the gold label, while $y_{token}, \hat{y}_{token}$ represents the edit distance label. We simply add the error detection and error correction losses as the final loss in the task.

Wrong Sentence Rewriting

For the wrong sentence rewriting task that contains all the above error types, we consider using a seq2seq model to cover the correction with a larger edit distance. The model architecture is shown in the right side of Figure 3. we finally chose BART(Lewis et al., 2019) as our backbone model due to its pre-training task of denoising. We believe that BART is more suitable for the task of text error correction than other models, because denoising and error correction are related tasks. Specifically, we used Shao et al. (2021)'s Chinese BART weights for model initialization.

Considering the amount of real training data, we still use the two-stage method of pre-training with pseudo data and fine-tuning with real data to train the error correction model. In the pre-training stage, we combined the previously constructed pseudo data (see in 3.1) and other open source semantic datasets (Sun et al., 2023) for training; in the fine-tuning stage, considering the data distribution, we integrated the Track2 and Track3 parallel corpus datasets for training.

We use the conventional way to train the seq2seq model. It should be noted that in order to avoid over-correction, we use greedy-search decoding in the inference stage.

4 Experiment

4.1 Data Analysis

We first counted the distribution of various errors in the data provided by Track1, which is used to guide our subsequent model training and pseudo-data construction guidelines.

The distribution diagram is shown in Figure 4. Through the analysis, no matter which source of the data set, Character-level errors account for the main part, and the metric of Character-level error contributes the most to the overall metrics. The result is in line with intuition, and Character-level error is also the most likely to occur and catch errors in composition writing scenarios.

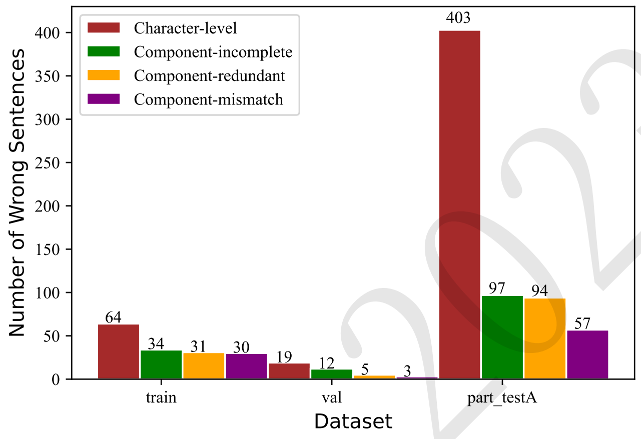

Figure 4: The distribution of each reference data set provided by Track1.

Specifically, the error of Character-level type accounts for a large proportion, followed by the error of Component Incompleteness and Component Redundancy, and the least is the type of Component Mismatch. Such a proportion also affects the amount of pseudo-data later to a certain extent.

4.2 Dataset

We present the source and number of other (unofficially provided) datasets we used in the datasets section. Overall, our model relies more on pseudo data constructed with Section 3.1. For Character-level error which has extensive open-source datasets, we use SIGHAN(Tseng et al., 2015) and CTC(Wang et al., 2022) as supplements. In addition, as mentioned in Section 3.1, we directly use the relevant parts of CSED(Sun et al., 2023) for some composition errors difficult to construct.

The amount of pseudo data we constructed for each category is shown in Table 1.

4.3 Metric

Referring to the requirements of the Task, we use the coarse-grained and fine-grained precision (P), recall (R), and F-score as metrics in Track1. And the F-score of character-level and sentence-level is used for Track2's Character-level correction model. Track3 integrates the edit distance label $F_{0.5}$, EM, Bert PPL, Levenshtein distance, BLEU-4, and BERT-Score as a reference.

4.4 Training Details

We generally use the AdamW optimizer and 128 max sequence length for model training. For Track1, we finally select the Chinese-electra-base model as encoder, we train four models for 50 epochs with a batch size of 32 and a learning rate of 5e-5 on the pseudo dataset, respectively. For Track2, we used Chinese-bert-base weight initialization. We pre-train on the pseudo dataset with batchsize 64, epoch 6, learning rate 1e-4, and fine-tune on the real dataset with batchsize 32, epoch 20, learning rate 5e-5. For Track3, we used the same settings as Track2, and submitted the best result on the validation set.

4.5 Validation Results

Due to time and submission mechanism factors, we mainly conducted ablation experiments on the setting of the error detection model.

Integrated Policy Validation

In order to verify our conjecture, we performed ablation experiments on the methods of directly predicting 14 error categories and model integration based on 4 coarse-grained error categories. As shown in the Table 3 below, the experimental results show that using four models to model various types of errors can reduce the task difficulty to a certain extent and improve the detection ability.

	Coarse-grained Error			Fine-grained Error			
method	**P**	**R**	**F**	**P**	**R**	**F**	**score**
single model	**35.41**	27.96	31.24	19.77	13.36	15.94	23.59
model integration	35.34	**68.51**	**46.63**	**21.16**	**36.45**	**26.77**	**36.70**

Table 3: Validation results using a single model and model integration method. For fair comparison, we used the Bert-base model for experiments.

Backbone Model Selection

Besides, we tried two transformer-based backbone networks, BERT(Devlin et al., 2018) and ELECTRA(Clark et al., 2020), for sequence labeling tasks. The experimental results are shown in Figure 4, ELECTRA achieves better performance on error detection due to discriminative-based pre-training tasks.

	Coarse-grained Error			Fine-grained Error			
model	**P**	**R**	**F**	**P**	**R**	**F**	**score**
BERT-base	**35.34**	68.51	46.63	**21.16**	36.45	**26.77**	36.70
ELECTRA-base	34.21	**89.96**	**49.56**	18.70	**46.17**	26.62	**38.09**

Table 4: Results on the validation set using different backbone networks. The structure of the two models both adopt the method of integrating four coarse-grained models.

Threshold Hyperparameter Selection

In order to determine an appropriate threshold, we tuned the threshold hyperparameters on the validation set, and the experimental results are shown in Figure 5. Combining the macro and micro metrics, we finally choose 0.99 as the confidence threshold hyperparameter.

4.6 Testing Results

Our final score and ranking on each track are shown in Table 5 below. According to the official evaluation metric, we achieved the first place in all the tracks.

5 Related Work

Due to the complexity of the Chinese language itself, Chinese text error correction has always been a challenging task. For Chinese spelling error correction, Hong et al. (2019) utilizes a pre-trained language model to generate candidate words. Cheng et al. (2020) uses GCN to enhance the modeling of confusing

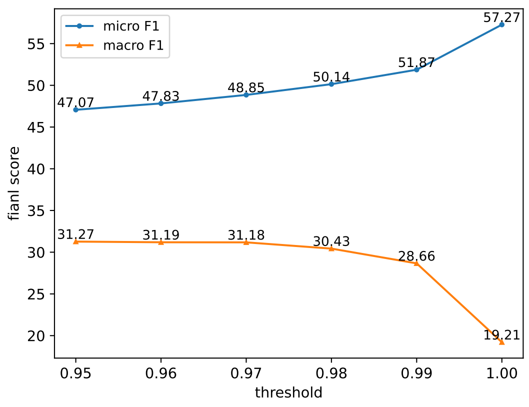

Figure 5: Illustration of threshold and detection metric on val under the optimal setting, which uses the ELECTRA-base backbone network and adopts model integration method.

Track	**Reference Metric**						**Score**	**Ranking**
1	Coarse-grained F1		Fine-grained F1		micro F1	macro F1	52.16	1
	micro	macro	micro	macro				
	56.7	42.81	47.62	16.46	52.16	29.64		
2	Detection		Correction		Identify	Correct	67.33	1
	char	sent	char	sent				
	62.28	55.86	62.76	40.02	74.22	60.44		
3	$F_{0.5}$	EM	PPL_{BERT}	LD	$BLEU_4$	$BERT_{Score}$	57.83	1
	45.81	17.34	2.91	1.91	89.85	97.60		

Table 5: Metrics and rankings on the official test sets for each track.

characters. Liu et al. (2021) proposes PLOME which incorporates pinyin information and font information to assist in error correction. And DCN(Wang et al., 2021) calculates the transfer matrix through the neural network, and generates more fluent error correction results through beam search.

For Chinese grammatical error correction, we continue to follow up the excellent methods of CGED. Wang et al. (2020) combines ResNet and Transformer structures for error detection. Luo et al. (2020) also uses GCN to model the syntactic dependency tree, thereby enhancing the error detection ability. Cao et al. (2020) proposes a feature-based gating mechanism, which can reduce the amount of training parameters of the model.

A major difficulty in Chinese grammar error correction is the lack of high-quality labeled data. In recent years, some scholars have also devoted themselves to the construction of high-quality grammar data sets. Zhang et al. (2022) relabeles and integrates the current CGEC dataset to build a multi-reference dataset that can evaluate the model more accurately. Xu et al. (2022), Sun et al. (2023) organizes manual annotation according to the exam questions of wrong sentences in primary and secondary school, and construct high-quality native Chinese grammar error datasets with fine-grained classification.

6 Conclusion and Future Work

This paper describes our detection and correction system on the CCL-2023 CEFE task, which includes token-level error correction and the implementation of two different paradigm GEC models. In all three tracks, the metrics of our model reached the first. In future work, we will continue to study more efficient grammatical error correction methods, such as the adaptation of the seq2seq model to error correction tasks, the improvement of the speed of autoregressive methods, and the utilization of LLMs like chatgpt, etc.

References

Yongchang Cao, Liang He, Robert Ridley, and Xinyu Dai. 2020. Integrating bert and score-based feature gates for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 49–56.

Wanxiang Che, Yunlong Feng, Libo Qin, and Ting Liu. 2020. N-ltp: An open-source neural language technology platform for chinese. *arXiv preprint arXiv:2009.11616*.

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. Spellgcn: Incorporating phonological and visual similarities into language models for chinese spelling check. *arXiv preprint arXiv:2004.14166*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu. 2019. Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 160–169.

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

Shulin Liu, Tao Yang, Tianchi Yue, Feng Zhang, and Di Wang. 2021. Plome: Pre-training with misspelled knowledge for chinese spelling correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2991–3000.

Yikang Luo, Zuyi Bao, Chen Li, and Rui Wang. 2020. Chinese grammatical error diagnosis with graph convolution network and multi-task learning. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 44–48.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: Tag, not rewrite. *ACL 2020*, page 163.

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, Melbourne, Australia, July. Association for Computational Linguistics.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Hang Yan, 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*.

Bo Sun, Baoxin Wang, Yixuan Wang, Wanxiang Che, Dayong Wu, Shijin Wang, and Ting Liu. 2023. Csed: A chinese semantic error diagnosis corpus. *arXiv preprint arXiv:2305.05183*.

Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. Introduction to sighan 2015 bake-off for chinese spelling check. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 32–37.

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.

Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. A hybrid approach to automatic corpus generation for chinese spelling check. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2517–2527.

Shaolei Wang, Baoxin Wang, Jiefu Gong, Zhongyuan Wang, Xiao Hu, Xingyi Duan, Zizhuo Shen, Gang Yue, Ruiji Fu, Dayong Wu, et al. 2020. Combining resnet and transformer for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 36–43.

Baoxin Wang, Wanxiang Che, Dayong Wu, Shijin Wang, Guoping Hu, and Ting Liu. 2021. Dynamic connected networks for chinese spelling check. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2437–2446.

Baoxin Wang, Xingyi Duan, Dayong Wu, Wanxiang Che, Zhigang Chen, and Guoping Hu. 2022. Cctc: A cross-sentence chinese text correction dataset for native speakers. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3331–3341.

Lvxiaowei Xu, Jianwang Wu, Jiawei Peng, Jiayu Fu, and Ming Cai. 2022. Fcgec: Fine-grained corpus for chinese grammatical error correction. *arXiv preprint arXiv:2210.12364*.

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. *arXiv preprint arXiv:2204.10994*.