# 鱼饼啾啾队CCL2022-Track4 比赛技术报告

吴修宇[1,2] 汤宸名[1,3] 吴云芳[1,3]
[1] 计算语言学教育部重点实验室（北京大学）
[2] 北京大学软件与微电子学院
[3] 北京大学计算机学院
{xiuyu_wu, wufy}@pku.edu.cn
tangchenming@stu.pku.edu.cn

## 摘要

本文描述了鱼饼啾啾队在第二十一届中国计算语言学大会（CCL-2022）汉语学习者文本纠错评测比赛第四赛道（即多参考多来源汉语学习者文本纠错）中提交的参赛系统。系统使用了seq2seq和seq2edit模型，并分别选择了合适的预训练模型进行参数初始化，并对模型的输入和输出进行了细致的处理，确保分词等处理过程中的噪声不会被视为模型所预测的修改操作。同时，我们设计了基于规则的加噪方式，通过对正确文本的加入噪音，生成大量可用于预训练的人工数据，更进一步提升了单个模型的性能。同时，我们在模型集成方面做了大量的尝试，并设计了更高效的集成方法。在最终的榜单中以51.02$F_{0.5}$排名第二，与第一名仅有0.13 的微弱劣势。

**关键词：** 语法纠错 ； 序列到序列模型 ； 序列标注模型 ； 数据增强 ； 模型集成

# The Technical Report of YuBingJiuJiu at CCL2022-Track4 Campaign

**Xiuyu Wu**[1,2] **Chenming Tang**[1,3] **Yunfang Wu**[1,3]
[1] MOE, Key Laboratory of Computational Linguistics, Peking University
[2] School of Software and Microelectronics, Peking University
[3] School of Computer Science, Peking University
{xiuyu_wu, wufy}@pku.edu.cn
tangchenming@stu.pku.edu.cn

## Abstract

In this paper, we describe our proposed system for CCL-2022 Track-4: Chinese Grammatical Error Correction with Multi-Reference Multi-Source Evaluation Dataset. In this contest, we combine both seq2seq and seq2edit models. We firstly select suitable pre-trained models for initialization. Secondly, we carefully design pre-processing and post-processing to ensure that those edits brought by vocabulary and tokenizer will not be considered as the correction operation predicted by the model. Meanwhile, we propose rule-based method to generate huge amount of pseudo data, which is utilized for pretraining to improve the performance of a single model. We also design more effective ensemble methods after a lot of experiments. We win the second grades with a performance of 51.02 $F_{0.5}$ score, which is slightly lower than the first team by 0.13 points.

**Keywords:** Grammatical Error Correction , Seq2seq , Seq2edit , data augmentation , model ensemble

## 1 任务介绍

CCL-2022 Track4是一场公开的语法纠错竞赛，是第二十一届中国计算语言学大会汉语学习者文本纠错评测（CCL2022-CLTC）(王莹莹et al., 2022)所设置的赛道之一。该竞赛使用的评测语料为苏州大学、阿里巴巴达摩院联合发布了MuCGEC（Multi-Reference Multi-Source Evaluation Dataset for Chinese Grammatical Error Correction）多源多参考中文语法纠错评测数据集(Zhang et al., 2022)。比赛在阿里云天池平台举办。

语法纠错任务具有重大的科研和实用价值。成熟的语法纠错模型可以帮助外语学习者进行实时的错误纠正，甚至协助母语者进行工作中的文档校对。中文语法纠错任务一直也吸引着研究者的关注。例如基于Lang-8数据和HSK语料的NLPCC-2018中文语法纠错竞赛(Zhao et al., 2018)，以及每年举办的CGED中文语法检错竞赛(Lee et al., 2016; Rao et al., 2017; Rao et al., 2018; Rao et al., 2020; Zhang et al., 2022)，都吸引了大量研究人员的参与。

本次竞赛使用的MuCGEC测试集，是基于NLPCC-2018测试集、CGED数据集以及部分Lang-8语料所构建的新的测试数据。数据集中对于每条中文句子，均由多人进行纠错。在评测中，使用的是基于字的M2score(Dahlmeier and Ng, 2012)，避免了先分词后评测的情况下，分词器带来的干扰。

基于本文的神经网络模型，我们开发了一个汉语文本纠错系统[1]，用户输入一个包含错误文法的句子，系统将自动输出纠错后的正确句子。

## 2 系统介绍

### 2.1 Seq2seq模型

我们选择了seq2seq模型作为我们的基础模型之一。Seq2seq模型在给定输入句子$X = (x_1, x_2, ..., x_n)$和目标句子$Y = (y_1, y_2, ..., y_m)$的时候，会以$P(Y|X) = \prod_{i=1}^{m} P(y_i|y_{<i}, X)$的形式进行建模。相比于seq2edit模型，seq2seq模型在处理乱序错误上更具优势。我们使用了Transformer作为Seq2seq模型的主干网络，同时使用了BART-base-chinese[2]进行模型参数的初始化。我们使用fairseq框架[3]实现模型，并在此基础上进行了训练和测试。相比transformers库，fairseq框架内置了更优秀的参数初始化和训练技巧，使得fairseq框架下的模型结果要更优一些。

### 2.2 Seq2edit模型

Seq2edit模型也被选作基础模型之一。Seq2edit模型在给定输入句子$X = (x_1, x_2, ..., x_n)$和目标句子$Y = (y_1, y_2, ..., y_m)$的时，会首先计算编辑序列$E = (e_1, e_2, ..., e_n)$。如果依次按照编辑操作$e_i$对于字符$x_i$进行修改，则可以将序列$X$转化为序列$Y$。编辑操作主要包括以下几种:

| 标签 | 含义 | 种类 | 训练集中占比 |
|---|---|---|---|
| *Keep* | 保持当前字符不变 | 1 | 86.04% |
| *Delete* | 删除当前字符 | 1 | 4.23% |
| *Append_t* | 在当前字符后添加字符$t$ | 3856 | 5.24% |
| *Replace_t* | 将当前字符替换为字符$t$ | 4410 | 4.48% |

Table 1: Seq2edit模型中使用的编辑操作。

在模型实现方面，我们使用了GECToR中文版的开源代码[4]。根据前人的实验结果(Zhang et al., 2022)，我们也选用chinese-struct-bert-large作为预训练模型。

### 2.3 模型集成

我们对于多个模型的结果进行集成。基础的集成的方式为，首先将模型输出的句子转化为模型对于原句的修改操作$edit_i = (x_i, j_i, op_i)$，表示对原句从下标$x_i$到下标$j_i$的段落进行$op_i$的修

[1] http://www.chinese-pku.com
[2] https://huggingface.co/fnlp/bart-base-chinese
[3] https://github.com/facebookresearch/fairseq
[4] https://github.com/HillZhang1999/MuCGEC/tree/main/models/seq2edit-based-CGEC

改。在多模型的集成中，首先设定阈值$T$，当一组修改操作$edit_i = (x_i, j_i, op_i)$被大于等于$T$个模型所提出时，才会被采纳。我们在此基础上进行了一些扩展：

- 模型按操作加权：根据统计，seq2seq和seq2edit模型在不同类型错误的纠错上的表现结果不用。而且不同的预训练方式也会导致相同类型模型针对相同错误的纠错能力存在差异。因此我们对不同模型提出的不同类型修改操作设置不同的权重。当一组被提出修改操作的权重和大于等于$T$时才接受。例如seq2seq模型更擅长处理乱序错误，则seq2seq模型提出的调序修改的权重可以设为2。这样在$T = 2$的情况下，由任意一个seq2seq提出的调序修改也会被接受。

- 分层集成：我们发现，当一个seq2seq和一个seq2edit模型进行集成时，其结果相比任意单个模型，都有极大的提升。但当多个seq2seq和多个seq2edit模型集成时，相比一个seq2seq和一个seq2edit模型的集成结果的提升则非常小。我们认为这个现象有两个原因：一是因为任意两个模型的编辑操作都存在大量重叠，新的模型所带来的新的修改的数量是快速下降的。二是因为为了保准修改的准确率，增加新的模型后，阈值$T$必须提高，部分正确但频率或权重和小于$T$的修改操作就不再被采纳。因此，我们选用一个seq2seq和一个seq2edit模型作为一组，组内以$T = 2$进行集成，得到prec较高但recall较低的结果。随后多组的集成结果以$T = 1$进行集成，在略微影响prec的基础上可以较大地提升recall值。

模型集成的代码实现我们参考了官方提供的集成脚本[5]。

## 2.4 特殊处理

### 2.4.1 切句后预测

我们将句号，问号和感叹号作为句子的结束符。同时为了防止引号内部的段落被切开，导致切分后的句子中出现不成对的引号的情况。我们将成对引号内部的句号、问号和感叹号不视作句子结束符。对于输入的句子，我们按照结束符进行切分，将切分后多个的子句分别送入模型进行预测，将得到的多个子句的输出拼成最终的输出。

### 2.4.2 还原非纠错修改

由于语法纠错模型在评测时，依据的是输出句子相对于输入句子做出的修改。而模型处理过中，会引入一些并非纠错的修改。例如使用BERT系列模型时，受限于词表，部分非常用字（例如：媒妁之言的“妁”），中文的全角引号等标点符号等，都会被替换为[UNK]字符，英文字符也会被自动转为小写。

对此，我们采用了预处理和后处理两种应对方式。后处理是基于编辑距离，计算原始文本到模型输出文本的编辑过程。编辑操作包括增、删、改三类。对于修改操作，我们会比对原文部分（被修改的段落）以及模型输出的部分（修改后的段落），如果发现模型输出部分是[UNK]，或者原文和模型输出的部分为内容相同但大小写存在差异的英文，则会认为这个修改是模型处理中的噪音而并非模型的纠错。我们会将该修改操作进行还原，选用原文的部分替换模型输出的部分。

预处理则是在后处理基础上的补充。我们发现部分圆角标点符号被替换为[UNK]后，模型会自动预测为另一种标点符号。而标点错误本身也是语法错误的一种，因此很难判定标点被修改的原因，是模型认为原始标点需要被修正，还是模型在处理中将原始标点替换为了[UNK]后，在[UNK]处补上了一个标点，不方便用后处理的方式解决。我们最后采用的是将BERT的词表中部分[unused]字符替换为中文圆角标点，让这些标点在分词后不会替换为[UNK]字符。替换[unused]的标点包括中文圆角上下双引号、中文圆角上下单引号、省略号、扩折号这6个标点符号。

对于单一模型的结果，是先通过预处理和后处理还原非纠错的修改后，再与别的模型的结果进行集成。

[5] https://github.com/HillZhang1999/MuCGEC/blob/main/scorers/ChERRANT/rule_ensemble.py

## 3 训练数据

### 3.1 真实数据

我们采用了官方提供的从Lang-8语言平台收集的数据，以及github上公开的HSK数据[6]。Lang-8平台上，母语非汉语的学习者会询问，某句话是否带有语法错误。母语学习者会给予解答，并给出修改结果。HSK语料则来源于母语非汉语的外国人参加高等汉语水平考试（HSK高等）作文考试的答卷，由专业的老师对错误句子进行修改标注。Lang-8语料数量大但质量参差不齐，HSK语料则数据量少但质量高。在数据清洗中，我们首先剔除了没有错误的句子，以及按照官方要求剔除了需要过滤的句子。我们按第2.4.2节所述，修改了BERT模型的词表，随后采用BERT分词器对句子进行了分词。我们将HSK数据上采样5倍后与Lang-8数据混合。最终各数据集的统计如下：

| 数据集 | 数量 | 平均句长 | 平均句长（分词后） | 平均编辑距离 |
|---|---|---|---|---|
| Lang-8 | 1,091,758 | 20.26 | 20.16 | 5.34 |
| HSK | 95,358 | 31.29 | 31.30 | 2.45 |
| 混合 | 1,568,548 | 23.65 | 23.54 | 4.46 |

Table 2: 训练集的数据统计。其中平均句长和平均编辑距离均按照分词后的结果计算。

### 3.2 人工伪数据

我们还通过在正确句子上依据规则添加噪音的方式，生成了大量人工伪数据，用于模型的预训练。原始语料选用了中文文本分类数据集THUCNews[7]，其中包含了14类共计74万篇新闻文档。对于每一篇新闻文档，我们删去了标题，作者姓名等信息后，按句号进行分句，得到了约1200万句。我们从中随机抽取了500万句作为伪数据的原始语料。

我们参考了留学生容易犯的语法错误，设计了3种不同层级的添加噪音的方式：

- 句子级别：如果句子中出现了成对的连词，比如“因为...所以...”或者“首先...其次...”，以一定概率删去成对连词中的一个，或者整对删去。
- 词语级别：对于被选中的一个词语，以一定概率将其被替换为同义词、形近词，或者删去词语中的一个或多个字，或者在小范围内移动其位置。
- 字级别：对于被选中的一个字，以一定概率将其替换为同音字、形近字或者[UNK]字符，或者将其删除。

本次使用的加噪方式包括了删除、替换和乱序三种。没有考虑添加的加噪方式，是因为没能找到合适的规则，能让添加的词语融入整个句子中，而不显得非常突兀。对于一个句子，我们首先判断其是否含有成对的连词。如果含有成对的连词，则以50%的概率对该句仅添加句子级别的噪声。对于不含成对连词，或者未被添加句子级别噪声的句子，我们使用jieba分词器进行分词。对于连续的8个词组成的段落中，随机选择一个词或字，并加入词级别/字级别的噪声。具体的各类噪音的添加概率以及使用资源如表3所示：

## 4 实验

由于比赛期间每天有提交次数限制，部分模型仅在官方验证集上使用官方评测脚本进行了测试。仅重要的模型结果会提交到平台上，在2000条官方测试集上进行测试。同时由于时间紧迫，我们通常会一边改进模型，一边改进数据处理方法。因此，部分数据处理方法的对比试验可能是在不同模型上完成的，部分实验设置也并未用于最终的模型，在此就不再详述，请见谅。

[6] https://github.com/shibing624/pycorrector
[7] http://thuctc.thunlp.org/
[8] http://ir.hit.edu.cn/demo/ltp/Sharing_Plan.html
[9] https://github.com/Aopolin-Lv/ECSpell/blob/main/Data/confusion/n_gram.json
[10] https://github.com/Aopolin-Lv/ECSpell/blob/main/Data/confusion/soundConfusion.json
[11] https://github.com/Aopolin-Lv/ECSpell/blob/main/Data/confusion/shapeConfusion.json

| 级别 | 类型 | 概率 | 外部资源 |
|---|---|---|---|
| 句子级别 | 删除成对连词中的前一个 | 0.4 | |
| 句子级别 | 删除成对连词中的后一个 | 0.4 | |
| 句子级别 | 删除整对连词 | 0.2 | |
| 词级别 | 替换为同义词 | 0.5 | 哈工大同义词词林[8] |
| 词级别 | 替换为同音词 | 0.1 | 同音词混淆集[9] |
| 词级别 | 删去词中的某个字 | 0.1 | |
| 词级别 | 删除整个词 | 0.1 | |
| 词级别 | 将词往前或后移动1-3个位置 | 0.2 | |
| 字级别 | 替换为同音字 | 0.7 | 同音字混淆集[10] |
| 字级别 | 替换为形近字 | 0.1 | 形近字混淆集[11] |
| 字级别 | 替换为[UNK] | 0.1 | |
| 字级别 | 删除这个字 | 0.1 | |

Table 3: 各类噪音的添加概率以及所需的外部资源。

### 4.1 特殊处理实验

第2.4章节中各类特殊处理方法的实验结果如下表4所示。可以看出通过还原非纠错的修改，可以在几乎不影响recall的情况下，提升precision指标，从而达到提升$F_{0.5}$指标的效果。而切句后再预测的方式对于seq2seq和seq2edit模型均有较大提升。我们认为这是测试集和训练集的句长存在较大差异导致的。MuCGEC的测试集由1996条NLPCC-2018测试集，3125条CGED测试集和1652条Lang8数据组成，平均句长为38.5个字符(Zhang et al., 2022)。而根据表2的统计结果，训练集的平均长度仅为23.65个字符。我们还在平均字符长度为29.7的NLPCC-2018测试集上做了测试，发现分局预测的提升较小，约为0.5的$F_{0.5}$指标。因此分句后预测的操作，提升了在较长的测试样例的纠错效果。

同时我们还发现奇怪的现象。分句后预测对于seq2edit模型可以提升precision和recall两项指标，但对于seq2seq模型会提升recall指标但降低precision指标。我们暂未找到合理的解释，欢迎大家探讨。

| 模型 | 训练语料 | 特殊处理 | 验证集 | | | 测试集（第一阶段） | | |
|---|---|---|---|---|---|---|---|---|
| | | | prec | recall | $F_{0.5}$ | prec | recall | $F_{0.5}$ |
| BART | Lang8 + 增强语料 | 无 | - | - | - | 58.50 | 17.68 | 40.02 |
| | | 还原非纠错修改 | - | - | - | 59.21 | 17.69 | 40.29 |
| BART | Lang8 | 还原非纠错修改 | 34.89 | 15.01 | 27.58 | 49.49 | 22.30 | 39.79 |
| | | 还原非纠错修改+ 切句后预测 | 32.60 | 18.60 | 28.33 | 47.02 | 26.48 | 40.71 |
| GECToR | Lang8 + HSK | 还原非纠错修 | 30.68 | 18.90 | 27.28 | 43.27 | 28.05 | 39.03 |
| | | 还原非纠错修改+ 切句后预测 | 33.69 | 21.16 | 30.12 | 45.72 | 30.68 | 41.64 |

Table 4: 各类特殊处理对结果的影响。

### 4.2 变更训练集实验

我们尝试分别使用Lang-8数据集，Lang-8和HSK混合的数据集进行训练，并且尝试先在500万条人工数据上预训练5轮后，再使用Lang-8和HSK混合的数据集进行微调。实验结果如下表所示。

可以看出高质量的HSK数据集对于结果的提升非常显著。而使用人工数据进行预训练，虽然$F_{0.5}$值会有所下降，但recall得到的提升。对于后续的集成来说，等于引入了更多备选的修改操作。而集成的过程中可以通过设定投票阈值来保证最终结果的准确率，这时，更多备选的修改操作意味着更高的集成结果上限。我们也发现，虽然加入人工伪数据后，BART和GECToR在官方验证集上的$F_{0.5}$是下降的，但是GECToR在NLPCC-2018测试集上

| 训练语料 | 模型 | 验证集 | | | 测试集（第一阶段） | | |
|---|---|---|---|---|---|---|---|
| | | prec | recall | $F_{0.5}$ | prec | recall | $F_{0.5}$ |
| Lang-8 | BART | 32.60 | 18.60 | 28.33 | 47.02 | 26.48 | 40.71 |
| | GECToR | 29.83 | 14.02 | 24.34 | - | - | - |
| Lang-8 + HSK | BART | 42.21 | 20.45 | 34.80 | 54.88 | 27.91 | 45.99 |
| | GECToR | 33.69 | 21.16 | 30.12 | 45.72 | 30.68 | 41.64 |
| 人工数据预训5轮+ Lang-8 + HSK | BART | 40.18 | 20.73 | 33.83 | - | - | - |
| | GECToR | 31.45 | 22.97 | 29.29 | - | - | - |

Table 5: 不同训练集对结果的影响。

的使用NLPCC-2018官方脚本的评测结果有较大提升，$F_{0.5}$从39左右提升至41.04。感觉不同来源的语法纠错数据集之间可能也存在一定的性质差别。

### 4.3 模型集成实验与最终提交

我们首先使用不同的随机种子以及略有区别训练设置（例如：调整dropout比例，对整个字符直接做dropout，随机将一些输入的字符替换为[UNK]等）来获取具有不同输出的模型，随后在进行集成。

| 模型 | 训练语料 | 验证集 | | | 测试集（第一阶段） | | | 测试集（第二阶段） | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | prec | recall | $F_{0.5}$ | prec | recall | $F_{0.5}$ | prec | recall | $F_{0.5}$ |
| BART-1 | Lang-8, HSK | 39.05 | 22.15 | 33.88 | 51.29 | 30.59 | 45.18 | - | - | - |
| BART-2 | Lang-8, HSK | 38.89 | 21.51 | 33.48 | 51.58 | 30.15 | 45.16 | - | - | - |
| BART-3 | Lang-8, HSK | 41.73 | 20.16 | 34.37 | 55.56 | 28.45 | 46.67 | - | - | - |
| BART-4 | 人工, Lang-8, HSK | 40.18 | 20.73 | 33.83 | - | - | - | - | - | - |
| BART-5 | 人工, Lang-8, HSK | 38.57 | 20.5 | 32.79 | - | - | - | - | - | - |
| GECToR-1 | Lang-8, HSK | 33.10 | 21.87 | 30.02 | 46.04 | 32.20 | 42.40 | - | - | - |
| GECToR-2 | Lang-8, HSK | 32.95 | 21.78 | 29.88 | 45.48 | 32.03 | 41.96 | - | - | - |
| GECToR-3 | Lang-8, HSK | 33.69 | 21.16 | 30.12 | 45.72 | 30.68 | 41.6 | - | - | - |
| GECToR-4 | 人工, Lang-8, HSK | 31.45 | 22.97 | 29.29 | - | - | - | - | - | - |
| GECToR-5 | 人工, Lang-8, HSK | 36.76 | 20.68 | 31.82 | - | - | - | - | - | - |
| Emb-1 | - | 54.23 | 15.26 | 35.89 | 67.89 | 21.86 | 47.77 | - | - | - |
| Emb-2 | - | 51.18 | 15.33 | 34.87 | 66.28 | 22.57 | 47.78 | - | - | - |
| Emb-3 | - | 52.72 | 16.17 | 36.31 | - | - | - | - | - | - |
| Emb-4 | - | 52.44 | 15.74 | 35.76 | - | - | - | - | - | - |
| Emb-5 | - | 54.55 | 14.96 | 35.65 | - | - | - | - | - | - |
| Emb-n3t2 | - | 43.86 | 21.89 | 36.53 | 56.53 | 30.09 | 48.08 | - | - | - |
| Emb-n3t2-W | - | 44.98 | 21.53 | 36.93 | 58.13 | 29.55 | 48.71 | - | - | - |
| PipeEmb-1 | - | 50.38 | 16.88 | 36.07 | 64.47 | 25.30 | 49.23 | 66.64 | 24.04 | 49.20 |
| PipeEmb-2 | - | 48.81 | 19.24 | 37.33 | 62.61 | 26.96 | 49.51 | 64.36 | 25.71 | 49.48 |
| PipeEmb-3 | - | 47.74 | 19.54 | 37.04 | - | - | - | 64.76 | 27.36 | 50.85 |
| PipeEmb-4 | - | 47.20 | 20.64 | 37.54 | - | - | - | 63.17 | 28.83 | 51.02 |

Table 6: 不同的集成方式对结果的影响。由于伪数据预训练的耗时较长，我们选择了在第一阶段结束到最终提交的期间进行预训练。因此没有在第一阶段的测试集上进行提交测试。

- Emb-i: 由BART-i和GECToR-i的结果，按照第2.3节所述的基础方式进行集成，阈值$T$设定为2。
- Emb-n3t2: 使用BART-1、BART-2和GECToR-3的模型输出，按照第2.3节所述的基础方式进行集成，阈值$T$设定为2。
- Emb-n3t2-W: 使用BART-1、BART-2和GECToR-3的模型输出，按照第2.3节所述的模型按操作加权的方式进行集成，阈值$T$设定为2，seq2seq模型对于调序修改的权重设为2，其余模型的各类修改权重均为1。

- PipeEmb-1: 按照第2.3节所述的分层集成，使用Emb-1和Emb-2集成后的结果，以阈值$T = 1$进行再次集成。
- PipeEmb-2: 按照第2.3节所述的分层集成，使用Emb-1、Emb-2以及Emb-3集成后的结果，以阈值$T = 1$进行再次集成。
- PipeEmb-3: 按照第2.3节所述的分层集成，使用Emb-1、Emb-2以及Emb-4集成后的结果，以阈值$T = 1$进行再次集成。
- PipeEmb-4: 按照第2.3节所述的分层集成，使用Emb-1、Emb-2、Emb-4以及Emb-5集成后的结果，以阈值$T = 1$进行再次集成。

从实验结果可以看出，集成模型Emb-i相比于单个BART或者单个GECToR模型，precision指标有了大幅提升，而recall指标略有下降，最终在$F_{0.5}$指标上有了较大的提升。随后我们尝试使用三个模型（BART-1、BART-2和GECToR-3）以$T = 2$进行集成。其结果虽然precision和recall的比例与Emb-i模型的差别较大，但在$F_{0.5}$略有提升。在Emb-n3t2的基础上我们扩展出了按操作加权的Emb-n3t2-W集成模型。我们考虑到seq2seq模型更擅长处理乱序错误，因此对seq2seq模型提出的调序修改赋予更大的权重，即任意一个seq2seq提出的调序修改都会被接受。最终结果显示该方式也在$F_{0.5}$指标上略有提升。

PipeEmb-i模型对应的是第2.3节所述的分层集成。以PipeEmb-1为例，相比原始的Emb-1和Emb-2集成模型，PipeEmb-1集成模型在precision略有下降的情况下提升了recall指标。又因为基础集成模型的precision偏高而recall偏低，提升recall的边际效应较大，导致$F_{0.5}$指标有了显著的提升。对于precision下降的原因，我们统计分析后得出结论，是因为Emb-1和Emb-2模型做出的修改操作中重叠的部分的precision高于模型整体的precision，而各自独有的修改操作的precision则低于模型整体的precision。在对集成模型再集成的过程中，两个基础集成模型重叠部分的修改仅被记录一次，导致分层集成的precision结果下降。对比PipeEmb-1和PipeEmb-2模型可以看出，在集成了Emb-1和Emb-2的基础上进一步集成Emb-3时，precision进一步下降，recall进一步提升，最终的$F_{0.5}$指标仅有微小的提升。我们尝试过在PipeEmb-2基础上再集成新的一组BART和GECToR模型的集成结果，结果是precison下降、recall提升后，$F_{0.5}$指标开始出现下降的情况。

而引入了使用人工伪数据预训练后的BART和GECToR的模型后，recall有了更进一步的提升。相比于PipeEmb-2模型，PipeEmb-3模型使用伪数据预训后的集成结果Emb-4替换掉了Emb-3，在precision变化不大的情况下，提升了2个点的recall。这表明使用人工伪数据进行预训练，可以使模型做出更多的修改，扩大了集成时的候选修改集合。但在PipeEmb-3的基础上再加入新的一组伪数据预训练的集成模型Emb-5后，仍然出现了precision下降，recall提升，$F_{0.5}$仅有轻微提升的现象。

以及在少数情况下，模型在验证集和测试集上的precision和recall不一定成正比（例如PipeEmb-2和PipeEmb-3模型）。可能是因为官方验证集每个错误句子仅有一个标准答案，而测试集上则是每个错误句子对应多个标准答案。以及官方的验证集主要使用的是Lang-8数据，而测试集的数据由多种略有差异的数据组成。

## 5 结语

本次比赛我们使用了seq2seq和seq2edit模型作为基础模型。首先进行了精细的前后处理，减少了处理流程中的噪音。在传统的投票集成基础上，扩展出了模型按操作加权和分层集成的方式。同时，自动生成了大量基于规则的伪数据用于模型的预训练，进一步提升了基础模型的效果。最终在以51.02的$F_{0.5}$得分获得了竞赛第二名。

## 致谢

项目支持：科技创新2030-“新一代人工智能”重大项目（2020AAA0106600），国家自然科学基金项目（62076008），国家自然科学基金重点项目（61936012）。

## 参考文献

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *HLT-NAACL*, pages 568–572. The Association for Computational Linguistics.

Lung-Hao Lee, Gaoqi Rao, Liang-Chih Yu, Endong Xun, Baolin Zhang, and Li-Ping Chang. 2016. Overview of nlp-tea 2016 shared task for chinese grammatical error diagnosis. In *NLP-TEA@COLING*, pages 40–48. The COLING 2016 Organizing Committee.

Gaoqi Rao, Baolin Zhang, Endong Xun, and Lung-Hao Lee. 2017. Ijcnlp-2017 task 1: Chinese grammatical error diagnosis. In *IJCNLP (Shared Tasks)*, pages 1–8. Asian Federation of Natural Language Processing.

Gaoqi Rao, Qi Gong, Baolin Zhang, and Endong Xun. 2018. Overview of nlptea-2018 share task chinese grammatical error diagnosis. In *NLP-TEA@ACL*, pages 42–51. Association for Computational Linguistics.

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, Suzhou, China, December. Association for Computational Linguistics.

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 *Proceedings of NAACL-HLT*, Online. 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 *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 439–445. Springer.

王莹莹, 孔存良, 刘鑫, 方雪至, 章岳, 梁念宁, 周天硕, 廖田昕, 杨麟儿, 李正华, 饶高琦, 刘正皓, 李辰, 杨尔弘, 张民, and 孙茂松. 2022. Cltc 2022：汉语学习者文本纠错技术评测及研究综述.