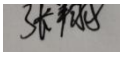


# 本 科 毕 业 设 计 (论 文)

| | | | |
|---|---|---|---|
| 学院(部) | 计算机科学与技术学院 | | |
| 题　　目 | 面向语音识别文本的纠错系统实现 | | |
| 年　级 | 2019 级 | 专业 | 计算机科学与技术 |
| 班　级 | 图灵班 | 学号 | 1927405126 |
| 姓　名 | 张翔 | | |
| 指导老师 | 李正华 | 职称 | 教授 |
| 论文提交日期 | 2023 年 5 月 6 日 | | |

# 苏州大学
# 本科毕业设计（论文）独创性声明

本人郑重声明：所提交的本科毕业设计（论文）是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本设计（论文）不含其他个人或集体已经发表或撰写过的研究成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人承担本声明的法律责任。

作者签名：　　日　期：2023-6-1

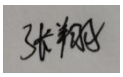
# 苏州大学
# 本科毕业设计（论文）使用授权声明

本人完全了解苏州大学关于收集、保存和使用本科毕业设计（论文）的规定，即：本科毕业设计（论文）的著作权以及文中研究成果的知识产权归属苏州大学。苏州大学有权向国家有关部门或第三方机构送交毕业设计（论文）的复印件和电子文档，允许毕业设计（论文）被查阅和借阅，可以采用影印、缩印或其他复制手段保存和汇编毕业设计（论文），可以将毕业设计（论文）的全部或部分内容编入有关数据库进行检索。

涉密设计（论文）□

本设计（论文）属　　在_____年___月解密后适用本规定。

非涉密设计（论文）■

论文作者签名：日　期：2023-6-1

导 师 签 名：日　期：2023-6-1

# 面向语音识别文本的纠错系统实现

摘要：

随着通信技术的发展，语音系统已经成为人们生活中不可或缺的一部分。智能识别、智能搜索等任务的兴起使得人们对语音识别结果的精确要求越来越高。在语音识别的过程中，由于说话人的背景环境嘈杂或者说话人的吞咽、拖拽、口音、方言等因素的影响，转录后的文本中通常存在较多的错误，这些错误会对下游任务性能造成直接的影响，进而影响用户的体验。面向语音识别文本的纠错系统旨在纠正语音识别文本中存在的错误，提高语音文本的准确率，从而优化语音搜索等任务的性能。

本文对语音识别文本纠错领域中一些经典模型进行了系统研究，并最终选取了GECToR 作为本研究的基础模型。在此基础上，我们引入了拼音特征。具体而言，我们采用了经过预训练的 BERT 模型来提取文本信息，同时对拼音信息进行编码处理。最终，我们将这两种编码相加并输入线性层以获得对应的标签。本文的具体工作如下：

首先，我们对该领域内的经典模型和主流模型进行了调研，并对他们的特点进行了详细的分析。

第二，我们采用预训练的 BERT 模型对源文本进行编码处理。接着，我们引入拼音信息，给赋予其一定的权重，将其融入到文本编码中以提高模型的纠错率。

第三，我们对语音数据集 AISHELL-1 进行处理，利用 ASR 和 2-fold 交叉验证方式来获取 ASR 错误文本，形成我们最终的语料集。

最后，我们利用数据集生成候选标签词表。

实验证明，融合了拼音信息的 GECToR 模型相较于传统的 GECToR 模型性能更优，拼音信息对语音识别文本纠错有着重要的意义，同时标签词表对模型的纠错能力也有重要的影响。

**关键词：**ASR 纠错，文本纠错，拼音信息

# Implementation of Error Correction System Oriented to Speech Recognition Text

Abstract:

With the development of communication technology, voice system has become an indispensable part of people's life. The rise of tasks such as intelligent recognition and intelligent search makes people have higher and higher requirements for the accuracy of speech recognition results. In the process of speech recognition, due to the speaker’s noisy background environment or the speaker’ s swallowing, dragging, accent, dialect and other factors, there are usually many errors in the transcribed text, and these errors will affect the performance of downstream tasks. Cause a direct impact, and then affect the user experience. The error correction system for speech recognition text aims to correct the errors in speech recognition text, improve the accuracy of speech text, and optimize the performance of tasks such as speech search.

This paper systematically studies some classic models in the field of speech recognition and text error correction, and finally selects GECToR as the basic model of this research. On this basis, we introduce the pinyin feature. Specifically, we use a pre-trained BERT model to extract text information while encoding pinyin information. Finally, we add these two encodings and feed into a linear layer to get the corresponding labels. The specific work of this paper is as follows:

First, we investigate the classic models and mainstream models in this field, and analyze their characteristics in detail.

Secondly, we use the pre-trained BERT model to encode the source text. Next, we introduce pinyin information, give it a certain weight, and integrate it into the text encoding to improve the error correction rate of the model.

Third, we process the speech dataset AISHELL-1, and use ASR and 2-fold cross-validation to obtain ASR error text to form our final corpus.

Finally, we utilize the dataset to generate a vocabulary of candidate tags.

Experiments have proved that the GECToR model incorporating pinyin information has better performance than the traditional GECToR model. Pinyin information is of great significance for text error correction in speech recognition, and the label vocabulary also has an important impact on the error correction ability of the model.

**Keywords**: ASR error correction, text error correction, pinyin information

# 目 录

**前　言 ..... 3**

**第一章 绪论 ..... 4**

1.1 研究背景及意义 ..... 4

1.2 研究现状 ..... 5

1.3 本文的主要工作及创新 ..... 6

1.4 本文的组织结构 ..... 7

**第二章　基础知识概述 ..... 8**

2.1 面向语言识别文本的纠错系统的基本概念 ..... 8

2.1.1 任务目标 ..... 8

2.1.2 评价指标 ..... 8

2.2 相关概念介绍 ..... 9

2.3 语音文本纠错模型概述 ..... 10

2.3.1 Seq2Edits ..... 10

2.3.2 FastCorrect ..... 11

2.3.3 PhVEC ..... 13

2.3 本章小结 ..... 14

**第三章 融合拼音信息的 GECToR 模型 ..... 15**

3.1 基础模型研究和改进的目的 ..... 15

3.2 LASERTAGGER ..... 15

3.2.1 背景介绍 ..... 15

3.2.2 模型介绍 ..... 15

3.3 GECTOR 基模型 ..... 16

3.3.1 背景介绍 ..... 16

3.3.2 原理介绍 ..... 17

3.3.3 模型框架 ..... 18
3.4 模型改进实现 ..... 19
3.4.1 数据处理部分 ..... 19
3.4.2 模型框架改进 ..... 19
3.5 本章总结 ..... 20
**第四章 实验过程和结果分析 ..... 21**
4.1 环境框架介绍 ..... 21
4.2 训练数据的生成 ..... 21
4.3 训练细节 ..... 22
4.4 实验结果与分析 ..... 23
4.5 本章总结 ..... 24
**第五章 总结与展望 ..... 25**
5.1 本文总结 ..... 25
5.2 工作展望 ..... 25
**参考文献 ..... 27**
**致谢 ..... 29**
**苏州大学本科生毕业设计（论文）任务书 ..... 30**
**外文文献资料 ..... 33**
**中文译稿 ..... 44**
**文献综述 ..... 62**
**中期检查表 ..... 73**
**答辩记录表 ..... 75**
**成绩评定表 ..... 77**
**检测报告 ..... 80**

# 前　言

随着自动语音识别技术的快速发展，其在人们的生活中的应用越来越普遍。随之而来的语音搜索，语音翻译等语音任务的兴起，使得人们对语音识别结果的准确性要求也日益提高。然而，由于说话人的背景噪音、语速、口音差异等音素的影响，语音转录后的文本常常存在着错误，这些错误会直接影响到后续语音任务的性能，进而影响用户的体验。为此，研究人员不断探索语音识别文本纠错的方法，以提高语音文本的准确率。

传统的序列到编辑方法（Seq2Seq）较早的应用于文本纠错领域并取得了一定的成果。然而，一些研究者开始关注该模型的局限性，并尝试将编辑方法（edit-based approach）应用于文本纠错任务，以提高模型的性能。值得注意的是，序列到编辑方法最初用于处理英文文本，后来也被迁移到中文文本领域。随着语音识别文本纠错的发展，研究员们通过融入拼音信息等方法来尝试不断优化序列到序列方法在本领域的效果。然而，序列到编辑方法却逐渐被忽视。但是语音识别文本输入和输出的高度重合性，导致序列到序列方法的训练资源有些浪费。因此，本文旨在将拼音信息融入序列到编辑方法，以提高纠错效果。

在本研究中，我们采用了基于序列到编辑思想的 GECToR 模型作为基础模型，并将拼音信息融合其中，取得了一定的效果。最终我们构建了面向语音识别文本纠错系统，并进行了实验评估。

本研究的结果表明，将拼音信息融入序列到编辑方法的方法在语音识别文本纠错任务中具有一定的优势。未来的研究可以进一步探索其他有效的纠错方法，并扩展到更广泛的语音识别应用领域。

# 第一章　绪论

## 1.1 研究背景及意义

随着人机交互技术的迅速发展，人们不再局限于传统的文本输入方式，语音输入逐渐成为人们使用设备的首选方式。语音系统的广泛应用已经渗透到各个领域。比如一些娱乐游戏中，玩家无法在玩的同时输入文本，通过语音识别系统可以更方便的表达玩家的想法，不靠谱的语音识别结果导致文本难以理解，甚至造成不必要的麻烦。因此，改进语音识别结果的准确性对于提升用户体验和满足下游任务需求具有重要意义。

在过去的时间中，大量的学者投入语音识别系统的研究中，现代语音识别最早可以追溯到 1952 年，Davis 等人设计了世界上第一个能识别 10 个英文数字发音的系统，随后，语音识别系统开始迈向了发展之路。在 1993 到 2009 年中，由于高斯混合模型(GMM)和隐马尔夫(HMM)模型的引入，语音识别一直处于 GMM-HMM 时代，语音发展进度缓慢。2006 年之后，随着深度学习尤其是循环神经网络(DNN)的兴起，语音识别框架转为 DNN-HMM，在这个时间段中语音识别正确率得到了显著的提升。而自 2015 年以来，“端到端”技术的崛起使得语音识别领域进入了快速发展阶段，通过使用端到端的方法训练更大更深的神经网络，进一步提高了语音识别的准确率。

尽管语音技术经过很长时间的发展，语音识别技术的鲁棒性仍是目前的研究重点。同时随着现今语音识别模型的性能不断提高，如何进一步改进语音识别模型来提高性能已经变得愈发困难。因此研究员们将注意力转到语音识别后的文本纠错领域中。近些年来，针对语音识别文本的纠正方法已经越来越成熟。人们通过将中文语法纠错等领域的模型转移到语音识别文本纠错领域并加以改进，取得了一定的成果。

同时，随着序列到序列(Seq2Seq)模型的发展，研究员们对其缺点进行了不断的弥补。尽管序列到序列方法在语音文本纠错领域取得了显著的进展，但序列到编辑（Seq2Edit）模型却逐渐被忽视。然而，序列到编辑方法具有高度的解释性和独特的优势，它能够通过对序列进行少量修改而得到目标序列。因此，基于序列到编辑的纠错系统的实现对于提升语音识别文本纠错的性能具有重要意义。

## 1.2 研究现状

语音文本纠错由于涉及语义表达、文本上下文信息融合、语句完整性检测和语言的搭配关系等内容，该领域的研究内容一直是自然语言处理研究中的难点。而语言的搭配关系又受到语义成分、人们的思维方式、不同地方的风俗习惯以及人们的认知情况等方面的影响，有很多学者在文本纠错领域中进行研究，取得了很多优秀的研究成果，推动了文本纠错研究的发展。

最初语音识别文本纠错领域采用了分阶段处理的方式，将纠错任务分为多个步骤多个模型，并通过串联这些模型的方式得到最终结果。这种方式的优点在于方便分析问题，能够针对各个问题进行重点突破。然而随着此方法不断发展，可读性不强、调试困难、性能损耗等缺点越来越明显。

随后，随着连续时间分类（CTC）方法的提出，端到端技术得到了发展。Graves 和 Jaitly 首先实现了基于 CTC 的端到端自动语音识别系统，获得了一定的效果。但是 CTC 贪婪解码的方式仍有较大的缺陷。2015 年，基于注意力机制的编码器-解码器模型首次被应用于自动语音识别系统中，极大的促进了此领域的发展。接着 Eric Malmi 和 Sebastian Krause 提出序列到序列方法，为机器翻译领域带来了巨大的科技变革。随着人们对此领域的深入研究，序列到序列方法不断演进。非自回归解码方式为序列到序列方法带来了更高的精确度，但也带来了时延高、解释性差以及对大量训练数据的需求等问题，使得在实际部署方面面临一定的困难。

2019，谷歌研究员独辟蹊径，提出一种新颖的序列到编辑方法，即 LaserTagger。该模型在保持精确度的前提下大幅提升了模型的解码速度，最终得到的标签序列也可以很好的解释错误的类型。该模型的提出为语音识别文本纠错领域带来了新的活力。

近年来，越来越多的研究人员投入到语音识别文本纠错领域的研究中。随着计算能力的增强和数据集的丰富，非自回归解码方式逐渐取代了自回归的序列到序列方法，并取得了令人满意的结果。特别是最近语音特征的融合，序列到序列方法在该领域取得了目前最先进的性能。

然而，序列到序列方法在重新生成源句子中的每个字符时似乎存在一定的浪费。相比之下，序列到编辑方法只需对错误的字符进行相应的变换，节省了计算资源。因此，序列到编辑方法仍然具有巨大的发展空间。

## 1.3 本文的主要工作及创新

本文利用融合拼音信息的序列到编辑方法对语音识别文本进行纠错。我们采用传统的 GECToR 作为我们的基础模型。在序列到编辑方法上引入拼音信息是我们与传统模型不同的地方，我们希望拼音信息的引入可能提高模型的纠错性能。

我们首先调研了近期面向语音识别的文本纠错相关工作，对目前该领域常用的各个模型进行对比分析，以此来对面向语音识别的文本纠错任务进行深入的了解。根据对各个模型的对比分析，结合语音识别文本同音字错误占比较多的特点，我们在 GECToR 模型的基础上引入了拼音信息，以此来帮助面向语音识别的文本纠错任务。接着，我们使用一定的方法在语音数据集获取我们的训练语料。再利用我们的数据集构造序列到编辑方法所需要的纠错标签，加入到纠错标签词表中。

最后我们在 AISHELL-1 语料上进行相关实验，最终的实验表明引入拼音信息能够有效帮助面向语音识别的文本纠错任务。

本文的主要工作及贡献如下：

(1) 我们首先介绍了面向语音识别文本纠错领域的一些评价指标。接着我们选取了此领域中一些经典模型并对这些模型进行了分析，最后选定了本文的基本模型。

(2) 分析本文的序列到编辑 GECToR 模型的原理以及框架，并对其模型进行复现。

(3) 对模型进行更改，将拼音信息融入到模型之中。

(4) 更新数据集的词表，更新标签，我们采用编辑距离的方式得到序列标签，将 min_count 设置为 3，即出现 3 次及以上就将此标签加入到词表中。

(5) 构造本文使用的数据集，我们在语音数据集上训练语音模型，利用 k-fold 强化学习，获得足够多的错误数据。我们将其中正确的数据去除掉，只留下错误数据以加强模型的纠错能力。

## 1.4 本文的组织结构

本文共分为五章，各章内容如下：

第一章：本章主要包括研究背景和意义、研究现状和本文的工作及贡献。接着介绍纠错领域的一些模型方法。最后介绍了本文的主要工作以及创新。研究背景和意义部分介绍了本领域的来源以及本系统的现实意义。研究现状介绍了当前此领域的现状。最后通过本文工作和贡献概述了本文的工作及实现思路。

第二章：本章包括本系统的任务目标、评价指标和本领域的经典模型。任务目标介绍了本系统需要实现的目标。评价指标解释了本领域一些评价指标。最后介绍和分析了本领域一些经典模型。

第三章：本章介绍了基础模型 GECToR，同时考虑到拼音信息在语音识别文本纠错领域的成功应用，我们在原模型的基础上引入了拼音信息。

第四章：本章我们首先介绍了获取数据集的方式，之后通过数据加强对训练集进行处理。最后通过实验结果证明我们方法的有效性。

第五章：本章主要是对本课题的总结和归纳，最后表达了对本课题未来的计划和展望。

# 第二章　基础知识概述

## 2.1 面向语言识别文本的纠错系统的基本概念

### 2.1.1 任务目标

在语音识别系统中，由于说话人的环境有点吵闹或者说话人有口音、语速过快、谈吐不清楚等情况导致翻译成的文本难以理解，文本中存在较多的错误。一般错误分为四种：冗余、漏词、如将“pian”识别为“ping”和“an”，形成冗余词。由于人的发音问题将“na”识别为“nei”,从而形成“内”等字。由于说话人语速快会将两个字识别为一个字，如“xi an”识别为“xian”。混合错误是总和，将“学不可以已”，翻译成“雪不可以”，“学”和“雪”同音以及将“yiyi”识别为“yi”[1]。这些错误往往会对下游任务(如语音搜索，语音查询等)的性能造成一定的影响。

相较于继续修改语音模型以提高识别文本的正确性，越来越多的研究员们将注意力转到语音识别文本纠错领域中，我们希望通过纠正语音识别后的文本来提高总体语音识别的正确率。

在现今语音识别文本纠错领域中，序列到序列方法已经取得了最高的性能。传统序列到编辑方法已经逐渐淡出人们视野。现今我们将其转移到语音识别文本纠正领域中，取得一定效果。结合此领域的发展，我们将拼音信息融入到序列到编辑方法之中以期待提高模型的纠错效果。在此领域中，我们使用本领域中一个经典的评价指标-词错误率(WER)来评判模型的纠错效果，我们需要通过纠错系统来降低文本的 WER，间接的提升语音识别系统的正确率。

### 2.1.2 评价指标

1）正确率(Accuracy)：一般来说，精确率越高越好，但是如果样本数据严重不平衡的，精确率会失真。如果正确数据在数据集中占比较高，那么整体的精确率就会很高，但不代表模型的纠正效果很高。

2）精确率(Precision，p)：表示正例中实际为正例的比例。

3）召回率(recall，r)：找出的真正的正例在所有正例中的比例。

一般情况下，精确率与召回率越高，表明模型效果越好。但是，精确率和召回率有些矛盾，总是存在精确率高而召回率低或召回率高而精确率低的情况，往往只有在一些简单的任务中，精确率和召回率都很高。

4）综合评价指标(f-Measure，F)：结合 p 值和 r 值计算得出，F 值计算公式如下：

$$F_\beta = (1+\beta^2)\cdot\frac{precision\cdot recall}{(\beta^2\cdot precision)+recall} \tag{2.1}$$

其中$\beta$越小，F 值越注重精确率，反之模型越注重召回率。本文采用 F0.5，相比于召回率，更注重精确率。

5）词错率(Word Error Rate，WER)：一项用于自动语音识别的重要指标，用来评价预测文本和标准文本间的错误率，WER 的特点是越小越好。

WER 的计算公式如下：

$$WER = \frac{S+D+I}{N} = \frac{S+D+I}{S+D+C} \tag{2.2}$$

其中 S 表示将预测文本转为标准文本发生的替换数量，D 表示预测文本转化为标准文本发生的删除数量，I 表示预测文本转化为标准文本发生的插入数量，N 代表目标文本中总的字数或者英文单词数量，C 代表预测文本中识别正确的字数。

## 2.2 相关概念介绍

现今语音识别文本纠错领域主要采用端到端模型。主流思想为序列到序列方法以及序列到编辑方法。从解码方式而言，模型又可以分为自回归模型和非自回归模型。

序列到序列(Seq2Seq):采用编码器和解码器结合的方法。是一种特殊的递归神经网络体系。利用源句子直接生成目标序列，其中两个序列可以不等长。

序列到编辑(Seq2Edit):也采用编码器和解码器的方法。利用源语句生成标签序列。每个标签的意义是将某个位置的字符进行替换删除或者复制等操作。最后通过一定的规则对源句子进行修改得到目标句子。

自回归模型：给定一个序列，使用深度神经网络对序列里面的每一个变量进行条件概率建模，每个变量的条件概率依赖于当前变量在序列里的所有前置变量。自回归

符合人们单向阅读的直觉和思维方式。采用迭代循环的方式，通过编码器对源语句进行编码后输至解码器，然后解码器根据源语句和上一步预测的目标端语言单词来预测下一个单词，一步步的输出最后的句子。模型可基于注意力(Ghazvininejad[2]等人)、BLSTM(Mani[3]等人)等。

非自回归模型：非自回归模型假设假设每个字都是独立的，相较于自回归模型，非自回归模型在长句子处理的速度上有着很大的优势，相比于一个一个递归循环解码，非自回归模型一次性并行生成句子中所有词语。但是非自回归模型由于没有很好的考虑字符间的条件依赖，正确率要稍微低一些。针对以上情况，现在工作提供了很多解决方法[4]。

## 2.3 语音文本纠错模型概述

### 2.3.1 Seq2Edits

1）模型介绍：

此模型[5]由 Marjan Ghazvininejad 等人于 2019 年提出。它不是直接预测目标序列，而是预测 N 个编辑操作的序列。每个编辑操作 $(t_n, p_n, r_n) \in T \times N_0 \times V$ 是个三元组，表示用与可解释标签相关联的替换字符 $r_n$ 替换源语句中从位置 $p_{n-1}$ 到 $p_n$ 的动作 $t_n$ (其中 T 是标签词表)。

此外采用跨度级编辑，这样可以更容易捕捉局部关系，并使得表现形式更加紧凑且易于学习。句子中的一个错误往往是通过多个字来解决的，跨度级别的编辑操作保留了跨度范围内的语言建模信息。

模型的输出是一个三元组不是字符序列。输出的概率如下：

$$\mathrm{P(y \mid x) = P(t_1^N, p_1^N, r_1^N \mid x) = \prod_{n=1}^{N} P(t_n, p_n, r_n \mid t_1^{n-1}, p_1^{n-1}, r_1^{n-1}, x)} \tag{2.3}$$

其中具体的网络结构如图 2.1。

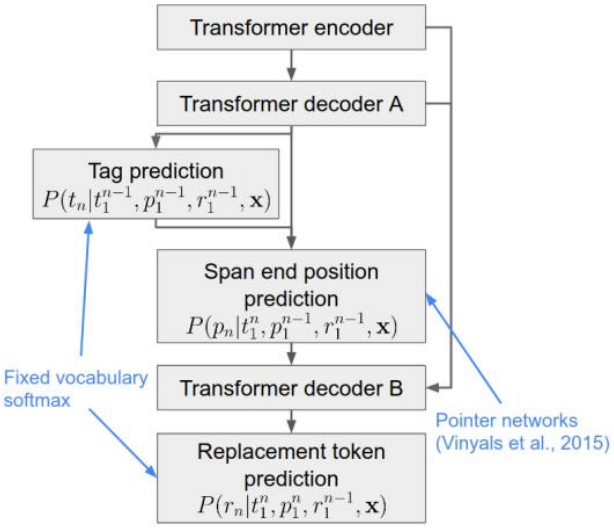



图 2.1 网络结构，图片来自文献[5]

在模型中将前一刻的预测反馈到 TransformerA 中,得到 $t_n$。再将此时输出输入到自注意力机制中来预测跨度结束位置，得到 $p_n$。

$$P(p_n \mid t_1^n, p_1^{n-1}, r_1^{n-1}, x) = \text{soft max}(\frac{QK^T}{\sqrt{d}}) \tag{2.4}$$

最后将之前的编码送到解码器 TransformerB 中来预测 $r_n$。可以将 TransformerA 和 TransformerB 看作是单个 Transformer 解码器层堆栈，将标签预测和跨度结束位置预测作为该堆栈中间的附加层添加。其中，将 transformerA 和 B 通过残差连接来促进学习。

2）模型优点：

在此之前的序列到编辑方法采用的字符级编辑，每次的编辑只关注当前的字符，忽视了周围字符间的联系。本文采用跨度级编辑操作，更好的考虑到了字符之间的联系，纠错效果更好。但是此模型并未考虑音素方面的一些信息，仍然需要一定的改进。

### 2.3.2 FastCorrect

1）模型介绍：

此模型[6]由 Leng Y 等人在 2021 年提出。在语音识别纠错文本中，源文本和输出文本的对应关系是单调的。所以可以在词级别来对齐源文本和输出文本，从而确定语

句中哪些是正确的、哪些是错误的。本模型中使用了编辑对齐算法，计算两个句子的编辑距离，得到数条编辑路径。从中选择“不变”最多的一条路径，尽可能少的去修改文本。如图 2.3。

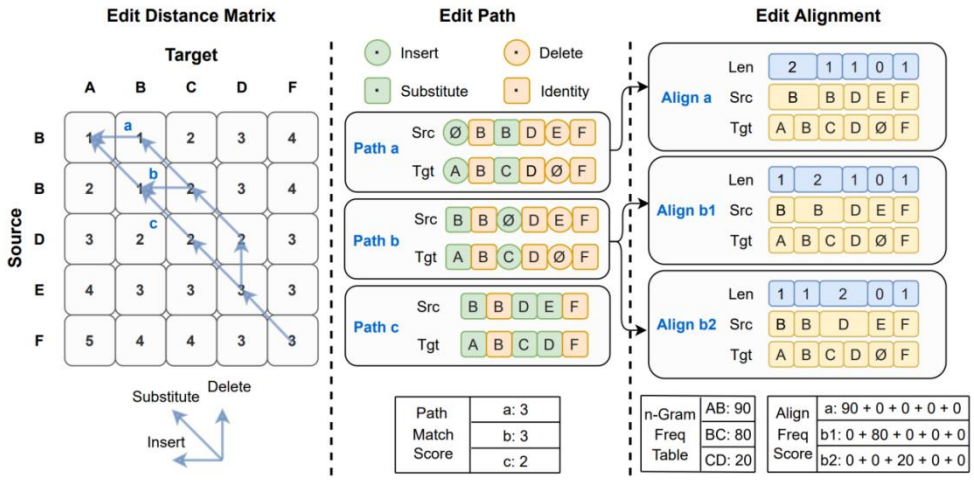



图 2.3 编辑路径示例，图片来自文献[6]

模型主要分为编码器、预测器和解码器三部分。Transformer 编码器用来学习源文本信息得到特征序列，这些特征将会输入到预测器中。预测器用来预测每个单词对应多少个单词，对应位置为 0 表示删除，1 表示替换或者不变，大于 1 表示需要用多个字来替换当前单词。解码器会同时利用编码器和预测器的输出。通过预测器调整最后输出的长度。再将编码器的信息输入到 Transformer 解码器中。解码器可以并行的一次性得到输出。

模型结构如图 2.4。

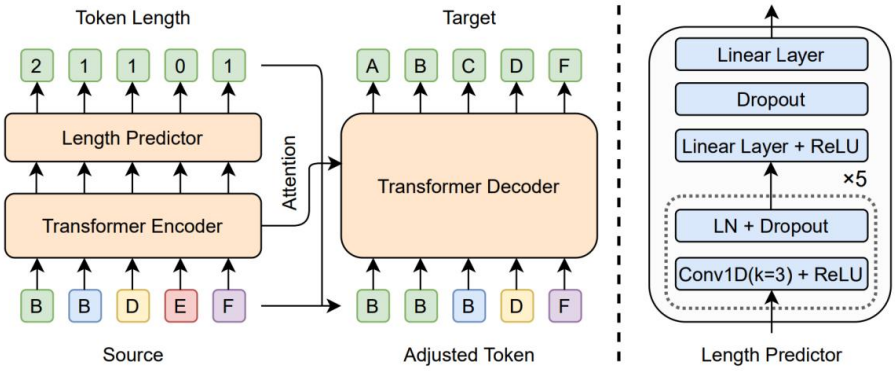



图 2.4 fast-correct 模型结构，图片来自文献[6]

2）模型优点：

此模型采取了编辑距离的方法，实现了对文本的变长预测。在此之前模型多是定长预测，但是一般语音文本可能出现删除增添等错误，不仅仅有替换错误，所以变长预测意义重大。同时本文成功的将非自回归模型应用于语音识别文本纠错领域，在保持较高精确率的同时提高了模型的解码速度。但是没有考虑到语音文本中的语音特征，仍可以对此模型进行修改。

### 2.3.3 PhVEC

1）模型介绍：

此模型[9]由 Zheng Fang 等人于 2022 年提出。不同于一些只支持定长纠错的模型[7][8]，此模型支持变长纠错。模型框架主要分为采用检查网络和纠正网络两部分[10]。

检测网络：此网络的目标是检查一个字符是否正确。可以将这个任务建模为序列标记。本文构建一个序列二进制分类器。用 1 表示正确的字符，用 0 表示错误字符。ground-truth 检测标签 C=$(c_1, c_2 \cdots c_n)$ 通过将源句子和目标句子与编辑距离匹配来预先计算，$c_i$ 表示位置 i 处字符被识别为正确字符 1 的概率。可以形式化如下：

$$\mathrm{p}_i = p(c_i^{'} = 1 \mid X) = soft\max(f_{dec}(E(e_w))) \tag{2.5}$$

$\mathrm{e}_w = (e_{x1}, e_{x2}, \cdots e_{xn})$ 是输入序列嵌入，E 表示基于 Transformer 的编码器，$f_{dec}$ 是将句子表示映射到二进制序列的全连接层，同时采用交叉熵函数训练模型。

纠错网络：使用 pypinyin 生成每个错误字符的拼音，在错误的字符后插入拼音特征，扩展源 token。之后通过 Transformer 和一个全连接层将其解码。其中很多时候源句子和目标句子的长度并不一样长，本文的处理方法是：通过发音对齐重复添加一些符号。本文将错误的拼音 token 和正确的进行比较，根据最长公共子字符串(CLS)算法进行对齐，如图 2.5。鉴于 $X_{\mathrm{cor}}$，纠错网络运用序列标注来预测纠正结果如下：

$$\mathrm{p}(y_i^{'} = v_j \mid X_{cor}) = soft\max(f_{crt} E(e_{cor})) \tag{2.6}$$

其中 $p(y_i^{'} = \nu_j \mid X_{cor})$ 是 $x_i$ 被正确纠正为 $\nu_j$（词汇表 $\nu$ 中的索引为 j 的 token）的概率。$e_{cor}$ 是 $X_{cor}$ 的 token 嵌入，$f_{crt}$ 是一个将 token 隐藏层映射到长度为 $|\nu|$ 的预测 logit 向量的全连接层。

最后采用联合学习同时优化两个模块.

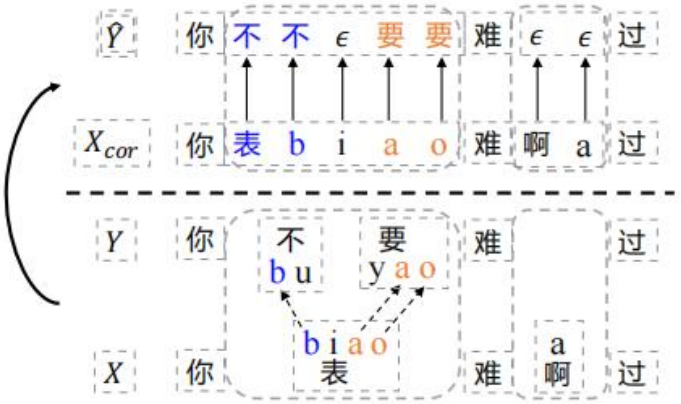



图 2.5 通过发音对齐的处理方法，图片来自文献[9]

2）模型优点

此模型融合了语音信息采用非自回归解码方式在中文语音文本上实现了变长预测。在中文语音识别文本纠错领域获得很高的效果，可以很好的处理语音文本中的一些变长错误。

## 2.3 本章小结

本章节主要讲解了面向语音文本纠错系统的概念及任务目标。接着介绍了此领域中的一些评价指标。然后对本文使用的语音识别文本纠错领域的一些概念进行了解释。最后本文调研了此领域一些经典的模型并从中选出 3 个进行了介绍。我们对模型的方法和框架进行了概述，并评价了模型的优缺点。

# 第三章　融合拼音信息的 GECToR 模型

## 3.1 基础模型研究和改进的目的

序列到序列方法通过源文本直接生成目标序列，但是输入输出文本的高度相似性使得直观上看会浪费计算资源。自回归的解码方式虽然有着较高的正确率，但是解码速度较慢。而非自回归解码方式在降低精确率的情况下大幅提升了模型的解码速度。

相比之下序列到编辑的模型通过源语句得到相关标签，再通过相关标签利用某些规则对句子进行修改，此模型的可解释性较强。考虑到一般语音文本中输入输出高度重叠,此模型只对其中错误的字符进行修改，节省计算资源，并且解码速度取决于编辑次数而不是序列长度。

随着现今语音识别文本纠错领域迅速发展，非自回归解码方式的序列到序列方法逐渐占据主流，而序列到编辑方法逐渐淡出人们的视野。而本文我们采用序列到编辑方法，并引入拼音信息。我们希望拼音信息的融入会提升模型的纠错性能。

## 3.2 LaserTagger

### 3.2.1 背景介绍

考虑到语音识别文本输入和输出文本较为相似，此方法将文本生成转化为文本编辑的任务。主要通过“插入”、“删除”和“保留”这三个方法重构目标文本。与需要大量训练数据的序列到序列方法相比，本文模型在较小数据集上取得了优于序列到序列方法的效果。在推理的过程中，此模型的速度比同类型的序列到序列方法快 2 个数量级。速度以及正确率上的优势，使得此模型在当时更有吸引力。

### 3.2.2 模型介绍

LaserTagger[11]模型采用编码器和解码器的架构。

编码器采用 BERT-base 架构，由 12 个自注意力层组成，用来学习源语句中的信息。我们使用训练 BERT 模型初始化编码器。原始 BERT[12]模型中提供了一个简单的解码机制用来序列标记。本文提出一个更强大的自回归解码器：在 BERT 编码器之上运行单层 Transformer 解码器。每一步将先前的预测标签输入到编码器中依次解码。模型框架如图 3.1。

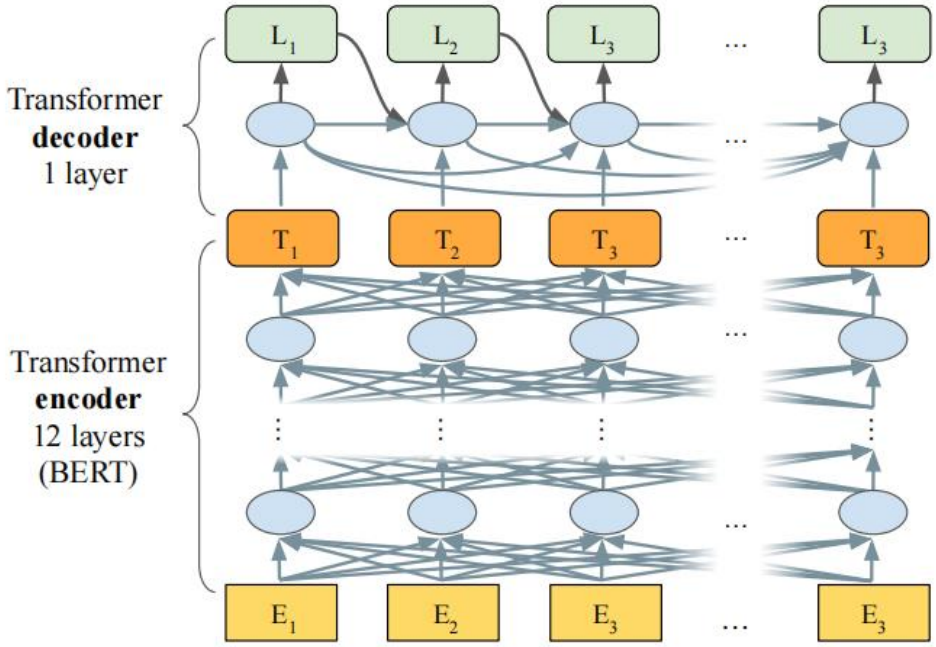



图 3.1 LaserTagger 模型框架，图片来自文献[11]

## 3.3 GECToR 基模型

### 3.3.1 背景介绍

基于神经机器翻译（NMT）[13]的方法已经成为一些语法纠错任务的主流方法。在这种方法中，我们将纠正任务当成翻译任务。最近，基于 Transformer 的序列到序列方法(Seq2Seq)在语法错误校正(GEC)中以及达到了 SOTA 性能。目前研究重点已转移到生成合成数据上，以对基于 Transformer-NMT 的语法错误纠正系统进行预训练。但是基于神经机器翻译的语法错误纠正系统有一些问题，这些问题使得这模型不太适合线上部署，不利于解决现实中的语音文本问题。第一无法并行解码，解码的时间就取决序列的长度，当对较长的句子进行解码时，耗时较长。第二训练序列到序列的模

型需要大量的数据，输入文本的长度不固定使得此模型需要更丰富的语料，而数据收集也有一定的难度。最后序列到序列的模型缺乏一定的可解释性。通常我们还需要借助其他的工具来分析句子中的错误类型。

基于上面的这些问题，所以 GECToR 模型[14]采用序列标签的方法。

### 3.3.2 模型方法

我们的模型不同于序列到序列方法，主要是生成一系列的序列标记，再通过一定的规则将源语句改成目标语句。这有点类似比较两个句子的不同点，找到一系列的编辑操作将错误句子转化为正确句子。我们赋予每个 token 一个标签，但是有时候需要对一个 token 进行多次操作，可每个 token 只对应一个序列标签，因此本文采取了迭代的方法，通过多次打标签的方法来对句子进行纠正。如表 3.2。

表 3.2 对一个句子进行修改的过程

| Iteration # | Sentence’s evolution | #corr. |
|---|---|---|
| Orig.sent | A ten years old boy go school | - |
| Iteration 1 | A ten-years old boy goes school | 2 |
| Iteration 2 | A ten-years-old boy goes to school | 5 |
| Iteration 3 | A ten-years-old boy goes to school. | 6 |

在表格中，第一次打标签，我们需要对 ten 和 go 进行编辑操作，将 ten 和 years 合并成了 ten-years、将 go 变成 goes。其中用连字符“-”将两个词合并且操作定义在前面的 token(ten)上。

接着第二次打标签，发现需要对 ten-years 和 old 以及 goes 进行操作。将 ten-years 与 old 合并成 ten-years-old，并且在 goes 后面加上 to。

最后一次打标签是在 school 后面加上“。”。

上面的那些编辑操作定义为对某个 token 的变换。如果词典由 5000 个词，其中就包含 4971 个基本变换和 29 个 g 变换。

基本变换包括与 token 有关的和与 token 无关的。与 token 有关的基本变换包括“$KEEP”(保持不变)和“$DELETE”(删掉当前字符)。与 token 相关的标签有“$APPEND_t1”和“$REPLACE_t2”，“$APPEND_t1”表示在当前的 token 后面加上 t1，

t1 不是需要所有词，因为有些词很少会出现遗漏。“$REPLACE_t2”表示用 t2 将当前的词替换掉。

g 变换对应一些特殊的情况。本方法一开始针对英文。英文中有很多时态和单数复数的变换，如果我们将单复数等词当成另一个词，那么会产生大量的词，这样模型也无法学习到时态等的变化，上面例子中将“go”变为“goes”即 g 变化。因此 g 变换主要应对一些非正常变化。

### 3.3.3 模型框架

在训练模型之前，我们需要对源语句进行分词，不同的 Transformer 可以采用不同的分词器(Tokenization) 。我们可以在词的角度对文本进行处理，也可以在字符的角度对文本进行处理，这取决于具体需求。

我们的整体模型框架较为简单，由 BERT 模型和 2 个全连接层构成。预训练的 BERT 模型对源语句进行编码，接着将输出传入到线性层中。两个线性层一个负责检测错误，获得对应检测标签，表示句子正确或者错误。另一个负责纠错，获得对应的标签。最后的到标签后，我们可以利用规则对源语句进行修改。具体模型如图 3.2。

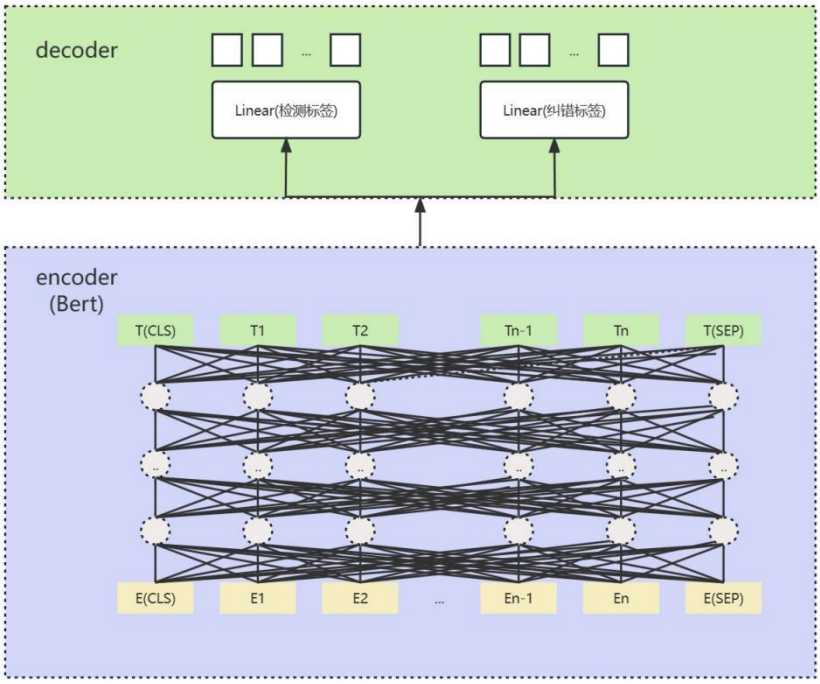



图 3.2 GECToR 模型框架，图片参考文献[14]

## 3.4 模型改进实现

### 3.4.1 数据处理部分

首先我们利用数据集构造对应的拼音字典。我们采用 pypinyin 生成训练集中的拼音，将其加入到拼音字典中。此外由于我们采用字符级编辑操作，我们将训练数据集中每句话进行分字，给每个字赋予一个纠错标签。

训练过程中，我们使用 pypinyin 得到源语句的拼音序列，再利用拼音字典将其转化为向量输入到模型中。也可以使用相关短语-拼音文件对句子进行最长字符串匹配得到拼音序列再转化为向量。

### 3.4.2 模型框架改进

我们采用将源句子信息和拼音信息分别编码的方法。源句子经过分词之后转化为向量，通过预训练 BERT 模型对源句子进行编码处理。拼音序列通过拼音字典转为向量之后经过 torch.nn.Embedding 对其进行第一次编码，接着通过一层线性层，线性层给拼音编码加上一定的权重。最后将两个编码输出相加输入到解码器中。最后通过线性层得到对应检测标签（表示句子是否错误）和纠错标签（用来修改源句子）。具体模型框架如图 3.3。检测网络和纠错网络的输入 e 表示如下。

$$e = e_{word} + W \cdot e_{pinyin} \tag{3.1}$$

W 为线性层权重矩阵，通过线性层我们可以对拼音信息进行加权。我们扩大了错误句子和目标句子的差异，加强了正确语句和目标语句的相似性。例如源句子“交易几乎停滞”与目标句子“交易几乎停止”。“停滞”中的“滞”为第四声，“停止”中的“止”为第三声，他们在字典中处于不同的位置。通过对拼音加权训练，结果的拼音信息会更靠近第三声的“止”，同时扩大“滞”于“止”的差异，使得纠正网络可以更好的识别出“滞”与“止”的不同，从而方便我们纠正错误。如果缺少拼音信息的约束，模型甚至能将“不要”（“buyao”）纠正为“表”（“biao”）。相较于语法纠错数据集，语音文本中含有较多类似上面的错误。拼音信息对于语法纠错影响较小，但是对于语音文本纠错意义重大。

最后我们对检测标签和纠错标签采用交叉熵函数进行训练。

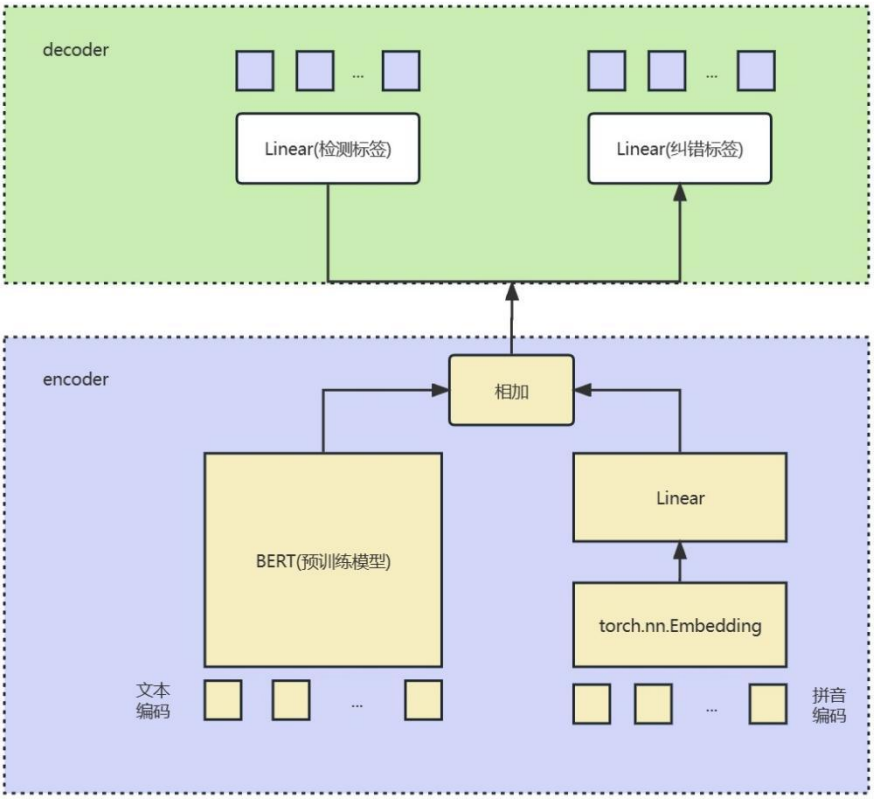



图 3.3 融合拼音信息的 GECToR 模型，图片参考文献[14]

## 3.5 本章总结

在本章中，我们首先介绍了采用序列到编辑思想的原因。接着介绍本文方法的基础模型，对其方法及框架进行了简单的分析。针对其未能很好利用拼音信息的缺点以及语音识别文本纠错领域文本的特点，我们提出了改进模型。最后我们介绍了模型的方法和思想以及改进后的框架。

# 第四章　实验过程和结果分析

## 4.1 环境框架介绍

本系统使用了 Allennlp[15]。这是一个基于 pytorch 的框架，面向深度学习中自然语言处理领域。此框架提供了一些最新的算法以及一些预训练模型。

Allennlp 对自然语言处理领域内的一些任务做了一定程度的抽象。它将流程封装成模块。我们可以只考虑模块间的逻辑，忽略多余的流程。

其中基本的一些模块有：

1) DatasetReader:此模块用来读取数据集，并最终将其转化为 Instance 类型。
2) Model:此模块包含一些 NLP 中主流模型。
3) Iterator:此模块用来构造迭代器，将读取的数据以迭代器的形式传入到模型中。
4) Trainer:此模块是模型的训练器，每个 epoch 后计算对应的 metric。
5) Predictor:此模块用来预测文本，利用原数据得到预测文本结果。

这些模块间的关联性较弱，需要较少的全局配置。我们可以单独处理每个模块，最后将模块整合。灵活使用此框架可以很大的简化 NLP 领域内的一些任务。

## 4.2 训练数据的生成

我们通过训练语音数据集来获得对应的训练文本。语音数据集有 AISHELL-1[16]、Magicdata-RAMC[17]等。AISHELL-1 训练集中包含 150 小时的普通话演讲，以及相应的转录本。主要包括财经、科技、体育等方面的新闻。该数据集提供相应的语音和对应的目标文本。

我们首先使用 WENET[18]工具箱，利用训练好的 Transformer+CTC[19]的 E2E 模型在 AISHELL-1 数据集上进行训练，将最终得到的语音文本结合语音目标文本分为 train、dev 和 test 三部分。通过语音模型得到的 train 数据集中错误数据比例太低。训练数据集中错误数据较少，语音纠错模型很容易进入收敛，模型缺乏纠错能力。

于是我们采取了下面的方法进行数据加强，丰富数据集中的错误数据。

第一种方法：对 train 数据集中数据进行了处理。随机取出数据集中的一些数据，由于语音文本的错误有很多来源于音调问题，我采用 pypinyin 得到句子的拼音信息。当一个人语速过快的时候，会将两个音调相似的字合成一个，于是随机将一些音调相似并且相邻的字去掉几个。对于语音文本中的同音字，得到拼音后我们通过 Pinyin2Hanzi 在候选字中选取字进行替换。但是由于替换较为随机，可能会导致数据集中错误间没有逻辑性，较为混乱。使用处理完的数据集进行了训练测试，模型的纠错效果一般，随机替换有时候过于牵强。

第二种方法：重新训练 Transformer+CTC 模型，采用 2-fold[20]对语音模型进行加强训练以期待得到更多错误的训练数据。dev 数据集和 test 数据集使用完整的 AISHELL-1 训练集进行训练得出。在最终得到的 train 数据集中，我们将其中的正确数据去除只留下错误数据以加强模型的纠错能力。

数据集概况如表 4.1。

表 4.1 AISHELL-1 数据集情况

| AISHELL-1 | Train | Dev | Test |
|---|---|---|---|
| 数据集规模 | 87633 | 14326 | 7176 |
| 错误数据 | 87633 | 8812 | 4600 |
| 正确数据 | 0 | 5514 | 2576 |
| 最长句子长度 | 44 | 35 | 37 |
| 最短句子长度 | 2 | 3 | 4 |
| 平均句子长度 | 14.82 | 14.32 | 14.59 |

## 4.3 训练细节

我们采用 Transformer+CTC 在 AISHELL-1 上进行训练，并采用 2-fold 在 train 数据集上获取对应的训练数据。再利用数据集构造我们的拼音字典。

接着为了将纠错任务转化为序列标注问题，我们将数据转化为序列标记，每个源字符对应一个标签。我们首先将每个字符映射到目标句子中的子序列。采用编辑距离来搜索目标字符最合适的子序列。对于列表中的每个映射，我们找到将源字符转换为

目标子序列的字符级转换。每个字符我们只保存一个标签，首先保留“KEEP”以外的标签。

此外我们还生成了对应的标签词表。不同的数据集有不同的特点。对应的标签表也有所不同。我们将源文本和目标文本进行对应处理，采用编辑距离对齐方式获得源文本转化为目标文本所需要的纠错标签，我们将 min_count 设置为 3，当对应标签出现 3 次以上我们就将对应的标签加入到标签词表中作为纠错标签。

在融合拼音信息的 GECToR 模型中，我们采用预训练 BERT 模型提取源文本的信息。模型一开始未饱和，我们首先将学习率设置为 1e-3 以加快模型的收敛速度。在经过两轮数据训练之后，模型预热完毕，再将学习率设为 1e-5 进行正常训练。

训练过程中，为了节省显存，我们将梯度累计次数设置为 2。我们采用交叉熵函数作为损失函数，采用 Adam 作为优化器进行优化。

## 4.4 实验结果与分析

我们在 AISHELL-1 数据集上运行了实验，并对融合拼音信息的 GECToR 模型和没有融合拼音信息的 GECToR 模型的纠正效果进行了对比，具体情况如表 4.2。

通过比较得知，拼音信息会使模型更好的理解语音识别文本，识别语音文本中的错误，从而提高模型的纠错效果。我们首先将拼音信息简单的通过 torch.nn.Embedding 进行编码后与文字编码相加，模型的纠错效果一般。之后我们在拼音编码后加上线性层赋予拼音特征一定的权重。通过训练我们可以适当调节拼音所占权重，效果得到了一定的提升，F0.5 提高了 9%左右。

表 4.2 模型结果对比

| | P | R | F0.5 |
|---|---|---|---|
| GECToR 模型 | 0.4071 | 0.3643 | 0.3977 |
| 融合拼音信息的 GECToR 模型 | 0.4866 | 0.5188 | 0.4927 |

此外我们测试了 test 数据集的 WER。经对比，我们发现融合拼音信息的 GECToR 模型有更好的效果。WER 降低了 3%，相较于原 GECToR 模型，额外降低了 0.5%左右。具体效果如表 4.3。

表 4.4 不同模型对应的 WER 变化

| | Test 原 WER | 纠正后的 WER |
|---|---|---|
| GECToR 模型 | 0.1074 | 0.0814 |
| 融合拼音信息的 GECToR 模型 | 0.1074 | 0.0767 |

值得注意的是，一开始我们直接使用了语音模型的转化后的文本，其中 train 中错误数据含量在 1%左右，约有 1 万条，而 dev 数据集错误数据含量为 40%，test 数据集错误数据含量为 50%。通过训练，模型纠错能力较差，正确数据过多、错误数据过少导致模型过早拟合。因此我们需要采取策略对我们的数据集进行加强，构造更多的错误数据进行训练，提高模型应对不同错误的能力。

结果表明标签词表对模型的纠正效果有着很大的影响。语音识别文本中含有较多的错误来源于语音特征问题，而中文语法纠错文本的错误类型不同于语音识别文本，我们需要生成对应的词表。我们还需要适当调整词表，如 min_count，我们将其设置为 3，即对应标签出现 3 次才会将其加入到标签词表中。

总之经实验表明，标签词表对模型的纠错效果有很大的影响，GECToR 最初应用于处理英文文本，作者为了应对英文中的特殊变换，在词表中加入了一些特殊变换。因此我们也可以通过方法适当丰富我们的词表。

同时利用拼音信息我们可以更好的区分正确字符和错误字符。如“停滞”和“停止”，“滞”为第四声，“止”为第三声。拼音信息的引入扩大了正确句子和错误句子的差别，模型可以更好的识别错误字符。最终实验表明，此方法能够提高模型的性能。

综上，通过实验证明，融合语音特征的序列到编辑方法在语音识别文本纠错领域有着较好的效果。

## 4.5 本章总结

在本章中，我们首先介绍了本文采用的环境框架。接着我们介绍了我们获取训练数据的过程。然后我们概述了训练过程中的一些细节，如参数设置、生成词表等。最后通过对比实验结果，我们发现我们的方法相较于原来的模型有着更好的性能并解释其原因。

# 第五章　总结与展望

## 5.1 本文总结

面向语音文本的纠错系统是自然语言处理领域一个非常重要的内容。语音系统在现今科技发达的时代使用的越来越多。这就导致语音系统的性能对很多的下游任务有着非常大的影响力。比如游戏里面的语音系统。当你想通过语音输入表达自己的意图时，错误的语音转化成的文本将会对本人造成一定的困扰。也比如语音搜索等，往往错误的语音识别会给你带来完全不一样的结果。综上所述，语音文本的纠错有一定的必要性。

本文首先介绍了此领域中的一些难点，接着介绍了本领域中的一些主要的思想。接着本文找到本领域中的一些比较经典重要的模型，介绍了他们的思想和基本的框架。本文重点分析了融合拼音信息的 GECToR 模型。作为序列到编辑思想中的一个比较经典的模型，本文分析了其主要思想、背景以及其主要的框架。接着本文在次模型的基础上提出了改进措施。

本文的工作及贡献主要如下：

（1）分析了本领域内一些传统和主流的模型并分析了其优缺点。并在此基础上挑选了本文的基础模型。

（2）分析了模型 GECToR 的优缺点，并将其原始模型复现了出来。

（3）在基模型的基础上提出了自己的改进方法，考虑到语音文本的一些特殊性，将拼音信息融合到模型中。

（4）对 AISHELL-1 语音数据集进行处理，采用多种方法对数据集进行加强。尽量构造多的错误数据，将 train 数据集中全部放入错误数据，防止模型过快的拟合，使模型更好的适应各种语音文本的错误。

## 5.2 工作展望

本文的面向自动语音识别文本的纠错系统相较于目前最先进的序列到序列方法

仍有一定的差距。本文采用的模型一开始时基于英文文本而提出，一定程度上没有考虑到汉字和英语的区别。同时拼音字典可以进行一定的增删丰富拼音词典。同时可以采用 Attention 机制进一步提取拼音信息。我采用 pypinyin 生成拼音信息并且构成拼音词典。此外我们可以训练神经网络模型结合部分标注数据对模型进行训练，获得更高质量的拼音信息。

同时可以对训练集再次进行数据加强，采取到足够多的错误数据进行训练。此模型还有词表，可以加强词表，适当丰富词表信息。

我们采用的预训练 BERT 模型对本文进行编码，也可以试试其他预训练模型进行比较。

本系统训练的数据集时 AISHELL-1 数据集，此数据集中没有标点符号。现实文本中含有较多的标点，我们也需要对含有标点的数据集进行实验。

# 参考文献


[1] Duan D, Liang S, Han Z, et al. Pinyin as a feature of neural machine translation for Chinese speech recognition error correction[C]//Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18. Springer International Publishing, 2019: 651-663.

[2] Ghazvininejad M, Levy O, Liu Y, et al. Mask-predict: Parallel decoding of conditional masked language models[J]. arXiv preprint arXiv:1904.09324, 2019.

[3] Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF models for sequence tagging[J]. arXiv preprint arXiv:1508.01991, 2015.

[4] Xiao Y, Wu L, Guo J, et al. A survey on non-autoregressive generation for neural machine translation and beyond[J]. arXiv preprint arXiv:2204.09269, 2022.

[5] Stahlberg F, Kumar S. Seq2Edits: Sequence transduction using span-level edit operations[J]. arXiv preprint arXiv:2009.11136, 2020.

[6] Leng Y, Tan X, Zhu L. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition[J]. Advances in Neural Information Processing Systems, 2021, 34: 21708-21719.

[7] Gu J, Bradbury J, Xiong C. Non-autoregressive neural machine translation[J]. arXiv preprint arXiv:1711.02281, 2017.

[8] Zhang R, Wang X, Zhang C, et al. Bstc: A large-scale chinese-english speech translation dataset[J]. arXiv preprint arXiv:2104.03575, 2021.

[9] Fang Z, Zhang R, He Z, et al. Non-Autoregressive Chinese ASR Error Correction with Phonological Training[C]//Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2022: 5907-5917.

[10] Zhang S, Huang H, Liu J, et al. Spelling error correction with soft-masked BERT[J]. arXiv preprint arXiv:2005.07421, 2020.

[11] Malmi E, Krause S, Rothe S, et al. Encode, tag, realize: High-precision text editing[J]. arXiv preprint arXiv:1909.01187, 2019.

[12] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[13] Luong M T, Manning C D. Stanford neural machine translation systems for spoken language domains[C]//Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign. 2015: 76-79.

[14] Omelianchuk K, Atrasevych V, Chernodub A, et al. GECToR--grammatical error correction: tag, not rewrite[J]. arXiv preprint arXiv:2005.12592, 2020.

[15] Gardner M, Grus J, Neumann M, et al. Allennlp: A deep semantic natural language processing platform[J]. arXiv preprint arXiv:1803.07640, 2018.

[16] Bu H, Du J, Na X. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline[C]//2017 20th conference of the oriental chapter of the international coordinating committee on speech databases and speech I/O systems and

assessment. IEEE, 2017: 1-5.
[17] Yang Z, Chen Y, Luo L, et al. Open source magicdata-ramc: A rich annotated mandarin conversational (ramc) speech dataset[J]. arXiv preprint arXiv:2203.16844, 2022.
[18] Yao Z, Wu D, Wang X, et al. Wenet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit[J]. arXiv preprint arXiv:2102.01547, 2021.
[19] Graves A, Fernández S, Gomez F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]//Proceedings of the 23rd international conference on Machine learning. 2006: 369-376.
[20] Anguita D, Ghelardoni L, Ghio A. The'K'in K-fold Cross Validation[C]//ESANN. 2012: 441-446.

# 致谢

在本次毕业设计过程中，我遇到了很多的问题，但是在张磊学长和李正华老师的耐心帮助下，最后我成功的解决了问题。在毕设的过程中，通过和老师学长的交流，我了解了许多东西，开阔了我的视野。我发现我的知识储备量太少了，甚至都还没有入门。很多的概念性的东西不清楚，同时缺乏一定的科研精神。我需要更加努力， 需要学会自己解决问题。遇到问题要自己想想是否有解决方法。

总之感谢张磊学长和李正华老师的耐心指导。非常感谢

# 苏州大学本科生毕业设计（论文）任务书

学院（部）：计算机科学与技术学院

<table>
<tr><td colspan="6">设计（论文）选题：面向语音识别文本的纠错系统实现</td></tr>
<tr><td>指导教师姓名</td><td>李正华</td><td>职 称</td><td>副教授</td><td>类 别</td><td>毕业论文</td></tr>
<tr><td>学 生 姓 名</td><td>张翔</td><td>学 号</td><td>1927405126</td><td>设计（论文）类型</td><td>应用型课题</td></tr>
<tr><td>年 级</td><td>2019</td><td>专 业</td><td>计算机科学与技术</td><td>是否隶属科研项目</td><td>是</td></tr>
<tr><td colspan="6">1、 设计（论文）的主要任务及目标<br>随着自动语音识别（ASR）技术的不断普及，对 ASR 精度的要求也越来越高。在 ASR 的过程中，有大量的转化错误需要处理，这些错误与传统的文本纠错不同，这里主要是由于同音字、连续等错误导致的。例如“你不要难过”，其中由于“不要”连续，可能被识别为“表”，原句被识别为“你表难过”。本设计吸收借鉴相关领域的最新研究成果，利用深度神经网络模型来实现面向语音识别文本的纠错系统。相比于传统的一些模型，我们将文本的拼音融入其中，希望通过融入拼音信息来进一步帮助对同音字错误的纠正，以此来提高对语音识别文本纠错的性能。通过本次毕业设计，不仅有利于深化对专业知识的理解，而且有利于提高学生运用所学知识分析、解决学术问题的能力。</td></tr>
</table>

2、 设计（论文）的主要内容

使用深度神经网络技术，搭建 ASR 文本纠错模型：

（1）找到训练的相关的数据集。

（2）找到相关模型，对其进行复现。

（3）构建自己的深度神经网络模型。

（4）在收集到的数据集上使用构建的模型进行训练。

（5）利用模型进行相关的测试，再适当调整参数。

3、 设计（论文）的基本要求

实现一个面向语音识别文本的纠错系统。

1.对于语音识别的文本，能够识别并纠正文本的错误。

2.对于文本内容能够获取到其对应的拼音信息。

3.融合语音识别文本信息和其对应的拼音信息，协助进行面向语音识别文本的纠错任务，来获取更好的性能。

4、 主要参考文献

[1] Hsin-Wei Wang,Bi-Cheng Yan,Yi-Cheng Wang and Berlin Chen.Effective ASR Error Correction Leveraging Phonetic.Asia-Pacific Signal and Information Processing Association Annual Summit and Conference.In APSIPA ASC,2022.

[2]Fan Zhang,Mei Tu,Song Liu and Jinyao Yan.ASR Error Correction with Dual-Channel Self-Supervised Learning.In ICASSP,2022.

[3]Yang Tao,Jian Zhang and Ziyue Niu.Pre-training with Phonetics and glyphs for Chinese Spelling Correction.2022.In ITOEC,2022.

[4]Zhao Guo,Yuan Ni,Keqiang Wang,Wei Zhu and Guotong Xie.Global Attention Decoder for Chinese Spelling Error Correction.In ACL-IJCNLP,1419–1428,2021.

[5]Yi-Chang Chen,Chun-Yen Cheng,Chien-An Chen,Ming-Chieh Sung and Yi-Ren Yeh.Integrated Semantic and Phonetic Post-correction for Chinese Speech Recognition.arXiv:2111.08400v1,2021.

[6]Jingyuan Yang,Rongjun Li and Wei Peng.ASR Error Correction with Constrained Decoding on Operation Prediction.arXiv:2208.04641,2022

[7]Dagao Duan,Shaohu Liang,Zhongming Han and Weijie Yang.Pinyin as a feature of neural machine translation for Chinese speech recognition error correction.In First Online,2019.

[8]Zheng Fang, Ruiqing Zhang, Zhongjun He, Hua Wu and Yanan Cao.Non-Autoregressive Chinese ASR Error Correction with Phonological

Training.In NAACL,2022.

5、进度安排

| | 设计（论文）各阶段任务 | 起 止 日 期 |
|---|---|---|
| 1 | 寻找相关资料，了解 asr 语音文本纠错相关知识。 | 2 月 1 日至 2 月 27 日 |
| 2 | 寻找论文和数据集，找寻相关方法。 | 2 月 28 日至 3 月 8 日 |
| 3 | 模型复现训练测试阶段。 | 3 月 9 日至 4 月 11 日 |
| 4 | 撰写毕业论文阶段。 | 4 月 12 日至 5 月 2 日 |
| 5 | 准备答辩阶段。 | 5 月 3 日至 5 月 9 日 |

# 外文文献资料

## Non-Autoregressive Chinese ASR Error Correction with Phonological Training

**Zheng Fang[1,2], Ruiqing Zhang[3]*, Zhongjun He[3], Hua Wu[3], Yanan Cao[1,2]**
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
[3]Baidu Inc. No. 10, Shangdi 10th Street, Beijing, 100085, China
[1,2]{fangzheng, caoyanan}@iie.ac.cn
[3]{zhangruiqing01, hezhongjun, wu_hua}@baidu.com

### Abstract

Automatic Speech Recognition (ASR) is an efficient and widely used input method that transcribes speech signals into text. As the errors introduced by ASR systems will impair the performance of downstream tasks, we introduce a post-processing error correction method, PhVEC, to correct errors in text space. For the errors in ASR result, existing works mainly focus on fixed-length corrections, modifying each wrong token to a correct one (one-to-one correction), but rarely consider the *variable-length* correction (one-to-many or many-to-one correction). In this paper, we propose an efficient non-autoregressive (NAR) method for Chinese ASR error correction for both cases. Instead of conventionally predicting the sentence length in NAR methods, we propose a novel approach that uses phonological tokens to extend the source sentence for *variable-length* correction, enabling our model to generate phonetically similar corrections. Experimental results on datasets of different domains show that our method achieves significant improvement in word error rate reduction and speeds up the inference by 6.2 times compared with the autoregressive model.

### 1 Introduction

Errors introduced by automatic speech recognition (ASR) usually affect the performance of downstream tasks such as phonetic search, speech translation, etc. In recent years, ASR error correction techniques have been proposed (Anantaram et al., 2018; Mani et al., 2020; Zhao et al., 2021; Leng et al., 2021) to refine the ASR output and correct errors in text space. Without loss of generality, we study Chinese ASR error correction in this paper.

Given the ASR result of an utterance, the goal of error correction is to generate a sentence with the wrongly recognized words corrected. Thus the ASR error correction can be modeled as a machine translation problem under conventional autoregressive sequence-to-sequence (Seq2Seq) framework (Guo et al., 2019; Hrinchuk et al., 2020; Mani et al., 2020). However, the autoregressive models suffer from inefficient decoding since the generation of each target token depends on previously generated characters (Figure 1 (a)). Furthermore, without considering the phonetic similarities, the method is prone to generate corrections with totally different pronunciation, as for the example, the error character ("表", *biao*) should be corrected into ("不要", *bu yao*) but the Seq2Seq model ignores phonological features and corrects it to a phonetically different correction ("不许", *bu xu*).

To speed up prediction, recent studies propose to take non-autoregressive (NAR) methods (Gu et al., 2018; Ren et al., 2020) for error correction, which generates target tokens in parallel. Most NAR approaches make fixed-length predictions that generate same-length output as the source input by directly tagging on the source text (Zhang et al., 2020). Some works further leverage phonological features to correct the ASR errors caused by similar pronounced characters (Zhang et al., 2021a; Cheng et al., 2020). Such methods have made great improvements in correcting simple errors, but **cannot** handle samples with different lengths of source and target, referred to as *variable-length error correction*. See Figure 1 (b) for illustration. The method successfully finds out the erroneous character ("表", *biao*), and substitutes it with its phonological feature "*biao*" for error correction. However, it eventually generates a wrong correction ("标", *biao*) under the constraint of fixed length prediction.

To overcome above constraint while preserving efficient prediction, NAR solutions for *variable-length* prediction are proposed (Leng et al., 2021; Gu et al., 2019). They first build a *length predictor* to estimate fertility, i.e., the number of target tokens

* This work was conducted at Baidu. Corresponding author: Ruiqing Zhang.

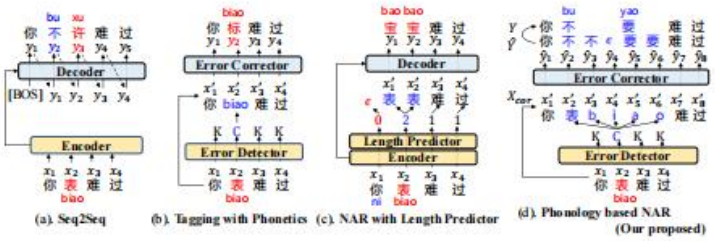



Figure 1: An example of Chinese ASR Error Correction. The source speech "*ni bu yao nan guo* " ("Don't be sad") is incorrectly recognized to "*ni biao nan guo* " ("Your watch is sad"). (a) The Seq2Seq model generates a fluent but incorrect sentence because it neglects the phonetic similarity. (b) The sequence tagging model with phonological features correctly detects the character with error ("K" means keep unchanged and "C" means to be correct), but because it only supports equal-length prediction, it fails to generate a fluent correction with equal length as the input. (c) The NAR model with a length predictor incorrectly predicts the number of target tokens corresponding to the source character "*ni*" as 0, which should be 1. Accordingly, it removes the character and generates an incorrect result. (d) The proposed PhVEC first detects problematic tokens, then leverages their phonological features to generate a *variable-length* intermediate result, and outputs the final result by reducing the repeated characters.

aligned to each source token, then up-sampling or dropping some source tokens accordingly for parallel correction. However, the method is fragile because incorrect length prediction will distort the meaning of the sentence and may directly lead to false prediction. See Figure 1 (c) for example, the length predictor produces a wrong alignment (0 2 1 1), indicating that the decoder will drop the first source character and generate two target characters for the second source character. According to the alignment, the intermediate result turns to be a meaningless sentence ("表表难过", means "watch watch sad") and leads to a wrong correction.

To address the issues mentioned above, we propose a novel NAR model named PhVEC (Phonology-based Variable-length Error Correction). The model incorporates the phonological features of error characters to enable *variable-length* prediction. Concretely, PhVEC contains a detection network and a correction network based on a pre-trained language model (Devlin et al., 2019; Sun et al., 2020). The detection network predicts the correctness of each token, and the correction network generates correction result. Instead of using a length predictor to predict the fertility for each source token, we insert the phonological tokens (Chinese Pinyin[1]) to the source sequence as placeholders after each detected erroneous character. During prediction, each source token (either the character token or pinyin token) can generate zero or one target token. With the guidance of the phonological token, the model will generate characters with similar pronunciations. As Figure 1 (d) shows, the pinyin token "*b*" produces target character ("不", *bu*) with the same consonant letter, and "*a o*" produces target character ("要", *yao*) with the same vowel letter. We delete the repeated characters in the final sequence.

We evaluate our methods on multiple datasets with varying degrees of ASR word error rates. Experimental results show that our PhVEC obtains significant improvement (over 10% word error rate reduction) on standard benchmarks compared with existing NAR-based ASR error correction methods at comparable speed. Even compared with the AR baseline Transformer and BART (Lewis et al., 2020) models, PhVEC can still have an 8.55% and 2.39% word error rate reduction while keeping a 6.2x speed-up.

## 2 Method

Chinese ASR error correction can be formalized as the following task. Given a speech recognition sequence $X = (x_1, x_2, ..., x_n)$ of $n$ Chinese characters, the goal is to correct it into another sequence of $m$ characters $Y = (y_1, y_2, ..., y_m)$. Note that the target sequence length $m$ does not have to be equal to the source sequence length $n$. There exist three types of ASR errors in transforming from $X$ to $Y$: substitution, deletion, and insertion.

As illustrated in Figure 1(d), our proposed ASR

[1]In this paper, we ignore the tone of pinyin, and the pinyin of each Chinese character can be represented by one to six English letters.

error correction model is composed of an error detector and an error corrector, both perform a NAR tagging. The error detector takes $X$ as input to predict the correctness of each token. The error corrector takes the combination of the source tokens and the phonological features of problematic tokens as input for correction. During error correction, each token of the extended input can be tagged to a Chinese character or a blank token "$\epsilon$", so that the model can support *variable-length* correction. Since different pinyin letters may generate the same character, we eliminate continuous repetitive characters in the final result. In this paper, we use Chinese pinyin as the phonological feature. The learning of PhVEC is conducted end-to-end, with the error detector and corrector optimized jointly.

### 2.1 Error Detection Network

The goal of the error detector is to check whether a character $x_i$ $(1 \leq i \leq n)$ is correct or not. We model this task as sequence labeling. We build a sequential binary classifier and use class 1 and 0 to label the problematic characters and correct characters, respectively. The ground-truth detection label $C = (c_1, c_2, ..., c_n)$ is pre-calculated by matching $X$ and $Y$ with edit distance, in which $c_i \in \{0, 1\}$. The prediction result of the error detector is represented by a sequence $C' = (c'_1, c'_2, ..., c'_n)$, and we use $p_i$ to denote the probability of token $x_i$ being predicted to class 1, which can be formalized as follows:

$$p_i = p(c'_i = 1|X) = softmax(f_{dec}(E(e_w))) \quad (1)$$

where $e_w = (e_{x_1}, e_{x_2}, ..., e_{x_n})$ is the token embedding of $X$, $E$ is a Transformer-based encoder and $f_{dec}$ is a fully-connected layer that maps the sentence representation to a binary sequence. To train the model, we adopt the following cross entropy loss function:

$$\mathcal{L}_{dec} = -\frac{1}{n}\sum_i [c_i \ln p_i + (1 - c_i)\ln(1 - p_i)] \quad (2)$$

### 2.2 Error Correction Network

After identifying the errors, we introduce the pinyin features of the incorrect characters and feed them into the correction network. Concretely, we use the tool PyPinyin[2] to generate pinyin for each erroneous Chinese character and insert pinyin tokens after the problematic token. The original

[2]https://github.com/mozillazg/python-pinyin

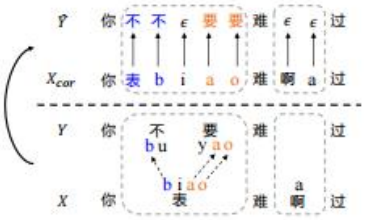



Figure 2: An illustration of generating $X_{cor}$ and $\hat{Y}$ from the training sample $(X, Y)$. We first label pinyin for incorrect characters in $X$ and their counterparts in $Y$ (bottom of the Figure). Then align the pinyin token according to the LCS algorithm (dashed arrows). $X_{cor}$ is obtained by inserting pinyin tokens after the incorrect characters in $X$. Finally, $\hat{Y}$ is generated according to the alignment. For NULL alignments, such as "了", we align a special token "$\epsilon$".

sentence $X = (x_1, x_2, ..., x_n)$ is thus rewritten as $X_{cor} = (x'_1, x'_2, ..., x'_t)$, where $t \geq n$, and $t - n$ is the number of pinyin tokens.

Given $X_{cor}$, the error corrector performs sequence labeling to predict the correction result $Y' = (y'_1, y'_2, ..., y'_t)$ as follows:

$$p(y'_i = \mathcal{V}_j|X_{cor}) = softmax(f_{crt}E(e_{cor})) \quad (3)$$

where $p(y'_i = \mathcal{V}_j|X_{cor})$ is the conditional probability that $x'_i$ is corrected to $\mathcal{V}_j$, the token with index $j$ in the vocabulary $\mathcal{V}$. $e_{cor}$ is the token embedding of $X_{cor}$ and $f_{crt}$ is a fully-connected layer that maps the hidden states of source tokens to their predicted logit vector of length $|\mathcal{V}|$. Note that, the parameters of the token embedding, the encoder $E$ and the correction network $f_{crt}$ are initialized by pre-trained language models (Devlin et al., 2019; Sun et al., 2020). We share the pre-training model parameters in the error detector and corrector to encode the Chinese characters and pinyin tokens in a shared space, which not only reduces the model size, but also makes the model yield better semantic representation.

The learning objective of the error corrector is to correct $X_{cor}$ to the golden correction $Y$. However, it is not an easy task because the lengths of $X_{cor}$ and $Y$ may not be the same. Therefore, we rewrite $Y$ to $\hat{Y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_t)$ which has the same length as $X_{cor}$. Concretely, $\hat{Y}$ is constructed from $Y$ with some tokens repeated and inserted according to pronunciation alignment. We compare the pinyin of the incorrect tokens of $X$ with their




Figure 3: The process of generating training samples. The characters marked in red from bottom to top represent the correct Chinese characters, the pinyin corresponding to the correct Chinese characters, the simulated pinyin disturbance, and the simulated error characters corresponding to the noisy pinyin, respectively.

correct counterparts of $Y$, and align according to the longest common substring (LCS) algorithm, as illustrated by the dashed arrows in the bottom of Figure 2. Then we rewrite $Y$ to $\hat{Y}$ according to the alignment.

The loss of the error corrector can be defined as follows:

$$\mathcal{L}_{cor} = -\sum_{i=1}^{t} \log(p(y'_i = \hat{y}_i | X_{cor})) \quad (4)$$

Our proposed pinyin alignment provides an explicit clue for the model to learn the correlation between pinyin and Chinese characters. As an alternative, we can also learn the alignment through Connectionist Temporal Classification (CTC) (Graves et al., 2006), which is widely used for *variable-length* alignments, such as in ASR (Audhkhasi et al., 2019), handwriting recognition (Bluche et al., 2014), etc. But we don't use CTC in our model because we assume that aligning according to the phonological features is reliable for ASR error correction, which will reduce the learning difficulty compared with learning a global alignment. We confirm our assumption in experiments.

### 2.3 Joint Learning and Inference

For each training sample $(X, Y)$, we construct its golden detection result $C$ and correction target $\hat{Y}$ first, then jointly optimize the two modules as follows:

$$\mathcal{L} = \lambda \cdot \mathcal{L}_{dec} + (1 - \lambda) \cdot \mathcal{L}_{cor} \quad (5)$$

where $\lambda$ is a trade-off parameter.

During the inference stage, we first detect the problematic characters and insert their pinyin into the original sentence, then predict the correction sequence $Y'$ by maximizing the probability of $p(y'_i | X_{cor})$ for each $i$. The final correction result is generated by removing the blank tokens $\epsilon$ and merging adjacent duplicate characters from the prediction of the error corrector, as shown at the top of Figure 1(d). Note that such post-processing is conducted only at locations where $X_{cor}$ is modified relative to $X$, while the characters predicted as correct by the error detector are kept unchanged in the final prediction.

## 3 Training Data Generation

We follow the common practice in error correction (Zhang et al., 2020; Takahashi et al., 2020; Leng et al., 2021) to synthesize the training corpus with simulated ASR errors. The simulated ASR results are generated by replacing the Chinese characters or words of clean sentences into problematic ones. Specifically, we generate the noisy text in three steps: (i) sampling some candidate characters and replace them with pinyin. (ii) adding noise to the original pinyin and generate new valid ones (iii) producing new Chinese characters or words based on the updated pinyin.

In the first step, the candidate words are obtained from a confusion set (Wang et al., 2019) that contains words prone to be mis-recognized. In the second step, we design three strategies of pinyin perturbation for simulating substitution errors, deletion errors and insertion errors, as shown in Figure 3. Note that when generating deletion and insertion errors, some random letters may be inserted to construct valid pinyin sequence. In the last step, we replace the noisy pinyin with corresponding word candidates, and select the sentence ranked highest by a n-gram language model as the final simulated ASR result.

## 4 Experiments

We carry out experiments on two Chinese ASR error correction datasets. We use word error rate (WER) and word error reduction rate (WERR) to evaluate our error correction performance.

### 4.1 Data Settings

We first train an ASR model on AISHELL-1 (Bu et al., 2017). The training set contains 150 hours of Mandarin speech, along with corresponding transcripts, mainly including news in Finance, Technology, Sports, etc. The trained ASR model then transcribes the speech to generate the paired data for evaluation, as listed in Table 1. AISHELL-1 dev and test sets contain 15 hours of speech, corresponding to 21K sentence pairs. MAGICDATA[3] contains 43 hours of Mandarin speech, including interactive Q&A, daily instructions, etc.

The training set of ASR error correction is constructed from 3 million web-crawled sentences, along with their noisy version with simulated errors. For the methods that only support fixed-length correction, the pseudo ASR input is generated with similar-pronounced substitution errors only. For the training data of methods that support *variable-length* correction, we sample 2.7 million samples from the fixed-length training data and generate simulated insertion and deletion errors for the remaining 0.3M sentences.

### 4.2 Training Details

Consistent with previous work (Leng et al., 2021), we use the ESPnet (Watanabe et al., 2018) toolkit to train an ASR model on AISHELL-1 training set. Conformer architecture (Gulati et al., 2020) and SpecAugment (Park et al., 2019) are also used to improve the ASR performance. For the ASR error correction network, we take "chinese-bert-wwm" (Cui et al., 2019) as a pre-trained model to initialize the encoders of our PhVEC and other pretraining-based methods. We set the learning rate to 5e-5 and the probability of dropout to 0.1. The loss balancing parameter $\lambda$ in joint learning is set to 0.5, and the AdamW (Loshchilov and Hutter, 2019) is utilized as optimizer. More details can be found in the Appendix A.

[3]https://openslr.org/68/

| Dataset | AISHELL-1 | MAGICDATA |
|---|---|---|
| Dev | 14,329 | 6,756 |
| Test | 7,176 | 15,131 |
| Avg Length | 14.42 | 9.78 |
| Type | News | Command |

Table 1: Statistical results on experimental test datasets.

### 4.3 Baselines

We compare our PhVEC with the autoregressive Transformer (Vaswani et al., 2017), BART[4] (Lewis et al., 2020) and recent state-of-the-art non-autoregressive methods as follows: **Levenshtein Transformer (LevT)** (Gu et al., 2019) supports *variable-length* correction by iteratively performing deletion, insertion and substitution under NAR framework. **FastCorrect** (Leng et al., 2021) implements *variable-length* correction with a length predictor that estimates the number of target tokens each source token should be converted to, then repeating/droping the source tokens to generate a variable length sequence as the input of error correction. **MLM-phonetics** (Zhang et al., 2021a) implements a fixed-length error correction by pre-training a language model with phonological features integrated. **BERT** directly fine-tunes the standard masked language model to generate fixed-length corrections. **BERT+CTC** implements *variable-length* error correction with a CTC layer built on top of BERT, which upsamples each source token twice and learns the source-target alignments automatically with a CTC loss.

### 4.4 Overall Results

The comparison results on two benchmark datasets are shown in Table 2. We observe that:

- Our proposed PhVEC outperforms all the other methods on the evaluation datasets. It achieves about 20% WERR for the four datasets with varying levels of ASR performance. PhVEC greatly exceeds the two existing methods supporting *variable-length* correction, LevT, and FastCorrect, and become the first method that surpasses the autoregressive Transformer and BART models, with the WERR improved by 8.55% and 2.39% on average, respectively.

[4]https://huggingface.co/fnlp/bart-base-chinese

| Method | AISHELL-1 | | MAGICDATA | | Latency(ms/sent) |
|---|---|---|---|---|---|
| | Dev | TEST | Dev | TEST | TEST |
| | WER↓ (WERR↑) | WER↓ (WERR↑) | WER↓ (WERR↑) | WER↓ (WERR↑) | GPU |
| No correction | 4.46 (-) | 4.83 (-) | 13.82 (-) | 13.51 (-) | - |
| Transformer | 3.80 (14.80%) | 4.08 (15.53%) | 12.09 (12.52%) | 12.19 (9.77%) | 149.5 (1×) |
| BART | 3.62 (18.83%) | 3.81 (21.12%) | 11.36 (17.80%) | 11.33 (16.14%) | 180.9 (0.8×) |
| LevT (Max iter=1) | 4.37 (2.02%) | 4.73 (2.07%) | 13.81 (0.07%) | 13.91 (-2.96%) | 54.0 (2.8×) |
| FastCorrect | 3.89 (12.78%) | 4.16 (13.87%) | - | - | 21.2 (7.1×) |
| BERT | 3.71 (16.82%) | 3.98 (17.60%) | 11.67 (15.56%) | 11.79 (12.73%) | 14.1 (10.6×) |
| BERT + CTC | 3.78 (15.25%) | 4.01 (16.98%) | 12.13 (12.23%) | 12.11 (10.36%) | 14.5 (10.3×) |
| MLM-phonetics | 3.64 (18.39%) | 3.90 (19.25%) | 11.59 (16.14%) | 11.68 (13.55%) | 23.7 (6.3×) |
| PhVEC (Ours) | **3.52 (21.08%)** | **3.62 (25.05%)** | **11.19 (19.03%)** | **11.04 (18.28%)** | 24.1 (6.2×) |

Table 2: Performance comparison of our method and other baselines on development set and test set. WER denotes word error rate (%), and WERR is word error reduction rate. We also test the inference speed of the correction models on NVIDIA V100 GPU and the test batch size is set to 1 sentence to match the online serving environment.

- Both PhVEC and MLM-phonetics introduce phonological features into the model but PhVEC performs better. We attribute this improvement to two aspects: one is its effective solution to *variable-length* errors, and the other is the effective use of pinyin splitting strategy. Different from Zhang et al. (2021a), we use pinyin features in letter granularity, instead of treating each pinyin as one token. This facilitates flexible insertion and enhances the correlation between similar pronounced characters and their corresponding phonological features.

- Pre-training significantly promotes correction performance. For the autoregressive models, the pre-trained BART has lower WER compared to the vanilla Transformer model. Moreover, the fine-tuned BERT achieves comparable performance with Transformer, indicating that strong language modeling will greatly facilitate NAR methods for error correction. Among the pretraining-based methods, PhVEC still performs the best, demonstrating the effectiveness of leveraging pinyin tokens for *variable-length* correction.

- Adding the CTC layer does not bring obvious advantages for error correction. In particular, BERT+CTC is inferior to BERT, and prones to generate incorrect alignments for the fixed-length correction samples. This might be because, for ASR error correction, the alignment between most words of the source and target is definite. BERT+CTC upsamples each source token twice and dictates the model to learn the alignment, which actually increases the difficulty of learning. It is worth noting that BERT+ CTC outperforms BERT on the *variable-length* error correction samples, which will be introduced later in Table 3.

- PhVEC accelerates to 6.2 times that of Transformer, and further reduces the WER by 8-10%. Even compared with the BART model, we still have 2-3% WER reduction and speed up the inference by 7.5 times, which proves the efficiency and effectiveness of PhVEC. Moreover, we want to emphasize that the original FastCorrect used NVIDIA V100 and P40 GPU to test the model delay respectively. To ensure the consistency of the results, we reimplement all baseline methods and test the inference speed on NVIDIA V100 GPU.

It is also notable that the training data of our method PhVEC includes 3M sentence pairs, which is consistent with that of Transformer, BART, LevT, BERT, and BERT+CTC, but FastCorrect and MLM-phonetics use additional 400M and 300M pairs sentences for pre-training, respectively.

### 4.5 Analyses

We further analyze the performance of our model on *variable-length* datasets and conduct ablation

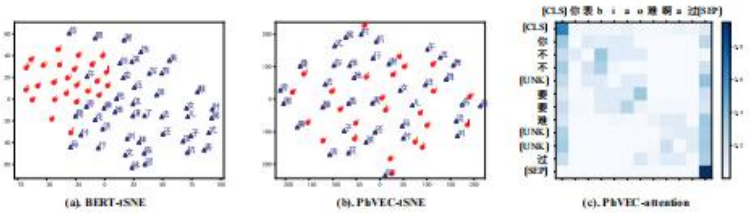



Figure 4: (a) and (b) are t-SNE visualization of pinyin letter and Chinese character embeddings, which correspond to BERT and PhVEC models respectively. (c) is the self-attention of the $3^{rd}$ encoder layer of the error corrector. The horizontal axis represents $X_{cor}$, the input of the error corrector, and the vertical tokens are the model output, in which [UNK] is for the blank token $\epsilon$.

| Method | AISHELL-1 | MAGICDATA |
|---|---|---|
| No correction | 15.02 (-) | 21.38 (-) |
| Transformer | 13.90 (7.5%) | 20.68 (3.3%) |
| BART | 12.08 (19.6%) | 17.30 (19.1%) |
| LevT | 14.34 (4.5%) | 23.11 (-8.1%) |
| BERT | 14.59 (2.9%) | 20.08 (6.1%) |
| BERT+CTC | 12.98 (13.6%) | 19.67 (8.0%) |
| MLM-phonetics | 14.48 (3.6%) | 20.12 (5.9%) |
| PhVEC (Ours) | **11.34 (24.5%)** | **16.06 (24.9%)** |

Table 3: The WER↓(WERR↑) evaluated on subsets that containing only the *variable-length* correction samples. The number of such samples in AISHELL-1 and MAGICDATA is 163 and 1490, respectively.

| Strategy | Example | AISHELL-1 | MAGICDATA |
|---|---|---|---|
| No correction | [表] | 4.83 (-) | 13.51 (-) |
| *OneToken* | [表][biao] | 4.08 (15.5%) | 12.11 (10.4%) |
| *Initial&Final* | [表][b][iao] | 3.91 (19.0%) | 11.68 (13.5%) |
| *Letters-only* | [b][i][a][o] | 3.85 (20.3%) | 11.59 (14.2%) |
| *Letters* (Ours) | [表][b][i][a][o] | **3.62 (25.1%)** | **11.04 (18.3%)** |

Table 4: The WER↓(WERR↑) of different phonological features evaluated on the AISHELL-1 and MAGICDATA test sets.

studies to dissect the factors affecting the effectiveness of our method.

#### 4.5.1 *Variable-length* scenario

To evaluate the effectiveness of our model in dealing with *variable-length* errors, we extract the samples containing insertion and deletion errors from the test sets of AISHELL-1 and MAGICDATA, and evaluate the above methods on this subset. As shown in Table 3, PhVEC shows significant advantages over other methods, achieving over 20% WERR on both datasets. Both fined-tuned BERT and MLM-phonetics perform weakly because of the inherent fixed-length limitation of their generation. The CTC layer brings significant improvement to BERT by enabling it with *variable-length* prediction, but there is still a large gap between its performance and that of PhVEC. The correction generated by LevT is not stable, even degrades the WERR by 8.1% WERR in the MAGICDATA subset. This might be because its length prediction result is inaccurate. The average length difference between the input and the golden correction is only 1.08, while the average length gap between the prediction of LevT and the golden correction is 1.42. This indicates that the prediction of LevT will produce more length differences than the original ASR input, resulting in its performance degradation.

#### 4.5.2 Different manners of leveraging phonological features

In our model, we split the pinyin of each Chinese character into tokens letter by letter and add them after each detected problematic character, e.g., rewrite the detected error character "[表]" (biao) to 5 tokens [表][b][i][a][o] in the intermediate *variable-length* result. Here we explore the impact of leveraging pinyin with different strategies: 1) *OneToken*: taking the pinyin of each character as one token as in Zhang et al. (2021a); 2) *Initial&Final*: divide each pinyin into an initial ([b]) and a final ([iao]), according to the phonological portion of Chinese; 3) *Letters-only*: remove the original problematic character from the rewritten sentence and use its pinyin for substitution. The first two settings focus on different pinyin granular-

ities, and the third one wants to check whether the original characters detected as errors are useful for error correction.

Table 4 shows that: (1) Compared with *Letters*, both *OneToken* and *Initial&Final* degrades the performance. This is because the granularity of the phonological features used in the two methods are coarser than that of *Letters*, which reduces the probability of aligning each phonological feature to more Chinese characters. (2) The *Letters-only* performs inferior to *Letters*, demonstrating the effectiveness of keeping the original characters in $X_{cor}$. This might be because, some wrong detection results tag "C" to some correct words. Removing these original words when adding pinyin tokens makes the model almost impossible to recover the correct ones. This always happens to rare words in some named entities.

#### 4.5.3 Characters Embedding and Alignment

To qualitatively examine whether PhVEC learns meaningful representations, we dive into the learned embedding and encoder for visualization. We first investigate whether the model learns the relationship between pinyin letters and Chinese characters. Concretely, we visualize the learned embedding of pinyin letter and Chinese character in a two-dimensional space by applying the t-SNE algorithm (van der Maaten and Hinton, 2008). Figure 4b shows that the embedding of Chinese characters and their corresponding pinyin letters get closer after training with PhVEC. Then we visualize the self-attention of the encoder in the error corrector. Figure 4c shows that the corrected words in the output pay much attention to their corresponding pinyin tokens, for example, "不(bu)" is highly aligned to "b", and "要 (yao)" paid much attention to "a" and "o", which indicates that the phonological features provide an obvious prompt for error correction.

## 5 Related Work

### 5.1 Chinese Spelling Error Correction (CSC)

As a close related area to ASR error correction, CSC has been widely explored in recent years. Zhang et al. (2020) proposes a soft-masked BERT model that first predicts the error probability of each character, and then uses the probabilities to perform a soft-masked word embedding for correction. As a remedy of soft-masked BERT, Zhang et al. (2021a) incorporates phonological knowledge into pre-training and proposes to fuse phonological feature in error correction. Cheng et al. (2020) builds a Graph Convolution Network on top of BERT, which reflects the phonological similarity among Chinese tokens. However, these methods are designed to produce corrections of the same length as the input, but incapable of handling *variable-length* correction that includes errors of substitution, deletion and insertion.

### 5.2 Autoregressive (AR) Error Correction

To correct *variable-length* errors, a large number of Seq2Seq AR models have been proposed. Zhang et al. (2019) uses a Transformer-based model for Chinese ASR error correction. Wang et al. (2019) incorporates a copy mechanism into Seq2Seq framework to copy the corrections directly from a prepared confusion set for the erroneous words. With the popularity of pre-training, Zhao et al. (2021) uses a pre-trained BART (Lewis et al., 2020) to initialize the correction network. Although these AR models are able to deal with various types of errors in ASR, they can not satisfy the latency requirements for online services, especially for some real-time scenarios like simultaneous translation (Zhang et al., 2021b).

### 5.3 Non-Autoregressive (NAR) Error Correction

NAR models are designed for fast generation speed compared with their AR counterpart by producing all tokens in a target sequence in parallel, which is widely explored in machine translation (Gu et al., 2019; Xu and Carpuat, 2021), ASR (Fan et al., 2021), TTS (Ren et al., 2019) etc. Recently, some studies (Gu et al., 2019; Leng et al., 2021) propose to apply NAR models to *variable-length* ASR error correction based on a length predictor, which first estimates the length of the target correction, then rewrites the input sentence by dropping or repeating some tokens according to the length estimated, finally performs a sequence labeling on the rewritten sentence to achieve correction. However, it is difficult to predict the target length of an incorrect sentence directly, even for humans, while the accuracy of length prediction is closely related to the performance of error correction.

## 6 Conclusion

In this paper, we propose a non-autoregressive Chinese ASR error correction network with phonolog-

ical training. Our method first detects the problematic characters, then adds the phonological features of them to adjust the input length, thus generating a *variable-length* sequence for error correction. The phonological features enable our model to produce similar-pronounced corrections, and support *variable-length* correction in a non-autoregressive mode. Experiments show that our method is superior to the autoregressive method while maintaining a 6.2x speed-up. As a future work, we plan to extend PhVEC to other languages and use corresponding phonological tokens to correct the variable length errors caused by pronunciation.

## Acknowledgements

This research is supported by the National Key Research and Development Program of China (NO.2017YFC0820700) and National Natural Science Foundation of China (No.61902394). We thank all authors for their contributions and all anonymous reviewers for their constructive comments.

## References

C. Anantaram, Amit Sangroya, Mrinal Rawat, and Aishwarya Chhabra. 2018. Repairing ASR output by artificial development and ontology based learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 5799–5801. ijcai.org.

Kartik Audhkhasi, George Saon, Zoltán Tüske, Brian Kingsbury, and Michael Picheny. 2019. Forget a bit to learn better: Soft forgetting for ctc-based automatic speech recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 2618–2622. ISCA.

Théodore Bluche, Hermann Ney, and Christopher Kermorvant. 2014. A comparison of sequence-trained deep neural networks and recurrent neural networks optical modeling for handwriting recognition. In *Statistical Language and Speech Processing - Second International Conference, SLSP 2014*, volume 8791 of *Lecture Notes in Computer Science*, pages 199–210. Springer.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. In *Oriental COCOSDA 2017*, page Submitted.

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. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 871–881. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese BERT. *CoRR*, abs/1906.08101.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics.

Ruchao Fan, Wei Chu, Peng Chang, Jing Xiao, and Abeer Alwan. 2021. An improved single step non-autoregressive transformer for automatic speech recognition. *CoRR*, abs/2106.09885.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006)*, volume 148 of *ACM International Conference Proceeding Series*, pages 369–376. ACM.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018*. OpenReview.net.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 11179–11189.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association*, pages 5036–5040. ISCA.

Jinxi Guo, Tara N. Sainath, and Ron J. Weiss. 2019. A spelling correction model for end-to-end speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 5651–5655. IEEE.

Oleksii Hrinchuk, Mariya Popova, and Boris Ginsburg. 2020. Correction of automatic speech recognition with transformer sequence-to-sequence model. In *2020 IEEE International Conference on Acoustics,*

*Speech and Signal Processing, ICASSP 2020*, pages 7074–7078. IEEE.

Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiang-Yang Li, Ed Lin, and Tie-Yan Liu. 2021. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition. In *35th Conference on Neural Information Processing Systems, NeurIPS 2021*.

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, ACL 2020*, pages 7871–7880.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net.

Anirudh Mani, Shruti Palaskar, Nimshi Venkat Meripo, Sandeep Konam, and Florian Metze. 2020. ASR error correction and domain adaptation using machine translation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 6344–6348. IEEE.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association*, pages 2613–2617. ISCA.

Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. 2020. A study of non-autoregressive model for sequence generation. *arXiv preprint arXiv:2004.10454*.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 3165–3174.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8968–8975.

Yujin Takahashi, Satoru Katsumata, and Mamoru Komachi. 2020. Grammatical error correction using pseudo learner corpus considering learner's error tendency. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, ACL 2020, Online, July 5-10, 2020*, pages 27–32. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 5998–6008.

Dingmin Wang, Yi Tay, and Li Zhong. 2019. Confusionset-guided pointer networks for chinese spelling check. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 5780–5785. Association for Computational Linguistics.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. Espnet: End-to-end speech processing toolkit. In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*, pages 2207–2211. ISCA.

Weijia Xu and Marine Carpuat. 2021. EDITOR: an edit-based transformer with repositioning for neural machine translation with soft lexical constraints. *Trans. Assoc. Comput. Linguistics*, 9:311–328.

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. 2021a. Correcting chinese spelling errors with phonetic pre-training. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 2250–2261. Association for Computational Linguistics.

Ruiqing Zhang, Xiyang Wang, Chuanqiang Zhang, Zhongjun He, Hua Wu, Zhi Li, Haifeng Wang, Ying Chen, and Qinfei Li. 2021b. BSTC: A large-scale Chinese-English speech translation dataset. In *Proceedings of the Second Workshop on Automatic Simultaneous Translation*.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 882–890. Association for Computational Linguistics.

Shiliang Zhang, Ming Lei, and Zhijie Yan. 2019. Investigation of transformer based spelling correction model for ctc-based end-to-end mandarin speech recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 2180–2184. ISCA.

Yun Zhao, Xuerui Yang, Jinchao Wang, Yongyu Gao, Chao Yan, and Yuanfu Zhou. 2021. BART based semantic correction for mandarin automatic speech recognition system. *CoRR*, abs/2104.05507.

## A Experimental Details

### A.1 Structure and parameters of ASR model

The ASR model is an end-to-end encoder-attention-decoder model with a 12-layer conformer encoder and a 6-layer conformer decoder, which is trained with cross-entropy loss on decoder output and an auxiliary CTC loss on encoder output. For the hyper-parameters of the ASR model, we take the beam search decoding with beam size to be 10, conformer kernel size to be 15, ctc weight to be 0.6, lm weight to be 0.3.

### A.2 Balance the objective of detection and correction

We explore the impact of the weighting strategy that balances the two objectives in fine-tuning. Table 5 presents the results of PhVEC in different values of hyper-parameter $\lambda$. Specifically, a larger $\lambda$ value means a higher weight on error detection, and the highest F1 score is obtained when $\lambda$ is 0.5.

| $\lambda$ | AISHELL-1 | | MAGICDATA | |
|---|---|---|---|---|
| | Dev(WER↓) | TEST(WER↓) | Dev(WER↓) | TEST(WER↓) |
| 0.2 | 3.61 | 3.75 | 11.42 | 11.37 |
| 0.5 | 3.52 | 3.62 | 11.19 | 11.04 |
| 0.8 | 3.74 | 3.83 | 11.55 | 11.47 |

Table 5: Impact of Different Values of $\lambda$.

# 中文译稿

## 基于语音训练的非自回归汉语 ASR 纠错方法

郑方，张瑞卿，何钟军，吴华，曹亚楠
中国科学院信息工程研究所
中国科学院大学网络安全学院
百度公司中国北京市上地 10 街 10 号，邮编 100085
{fangzheng,caoyanan}@iie.ac.cn
{zhangruiqing01,hezhongjun,wu_hua}@baidu.com

In NAACL,2022.

## 摘要

ASR(Automatic Speech Recognition,ASR)是一种高效而广泛使用的语音输入方法，它将语音信号转录成文本。由于 ASR 系统引入的错误会影响下游任务的性能，我们提出了一种文本空间错误的后处理纠错方法 PhVEC。对于 ASR 结果中的错误，现有工作主要关注于定长修正,即将每个错误的 token 修改为正确的 token(一对一修正)，很少考虑变长修正(一对多或多对一修正)。本文针对这两种情况提出了一种高效的非自回归(NAR)的中文 ASR 误差校正方法。与传统的 NAR 方法预测句子长度不同，本文提出了一种利用语音标记扩展源句子进行变长校正的方法,使我们的模型能够生成语音相似的校正。在不同领域数据集上的实验结果表明，我们的方法在降低单词错误率方面取得了显著改善,与自回归模型相比，推理速度提高了 6.2 倍。

## 1 介绍

自动语音识别(ASR)引入的错误通常会影响后续任务(如语音搜索、语音翻译等)的性能。近年来，ASR 纠错技术相继被提出（Anantaram et al.,2018;Aani et al.,2020;Zhao et al.,2021;Leng et al.,2021）来细化 ASR 输出并纠正文本空间中的错误。本文在不失一般性的前提下，研究了汉语 ASR 的纠错问题。

纠错的目标是在给定句子的 ASR 结果下，将错误识别的单词纠正生成一个句子。因此,ASR 纠错可以建模为传统的自回归序列对序列(Seq2Seq）框架下的机器翻译问

题（Guo et al.,2019;Hrinchuk et al.,2020;Mani et al.,2020）。然而，自回归模型存在低效解码的问题，因为每个目标令牌的生成依赖于先前生成的字符(图1(a))。此外，由于没有考虑语音相似性，该方法容易产生完全不同的语音校正，例如错误字符，错误字符（“表”，biao）应该被纠正为（“不要”，buyao）但是序列到序列的模型忽略语音特征，将其纠正为语音上不同的纠正结果（“不许”，buxu)。

为了加快模型的预测速度，最近的研究提出采用非自回归（NAR）方法（Gu et al.,2018;Ren et al.,2020）进行纠错，该方法并行的生成目标 tokens.大多数 NAR 方法进行固定长度的预测，通过直接标记源文本来生成与源输入相同长度的输出(Zhang et al.,2020)。一些工作进一步利用语音特征来纠正由相似发音字符引起的 ASR 错误(Zhang et al.,2021a;Cheng et al.,2020)。这样的方法在纠正简单误差方面有了很大的改进，但无法处理源和目标长度不同的样本，称为变长误差校正。如图1(b)所示。该方法成功地找出了错误字符(“表”，biao)，并将其替换为其语音特征“标”进行纠错。但最终在定长预测的约束下产生了错误的纠错(“标”，biao)。

为了克服上述约束，同时保持有效的预测，我们提出了变长预测的 NAR 解决方案（leng et al.,2021;Gu et al.,2019）。他们首先建立了一个长度预测器来估计生育能力，即对齐到每个 token 的目标 token 的数量，然后相应的上采样或者删除一些源 tokens，以进行并行纠正。然而，该方法是脆弱的，因为不正确的长度预测会扭曲句子的意思，并可能直接导致错误的预测。例如，参见图1(c)，长度预测器产生了错误的对齐(0 2 1 1)，表明解码器将丢弃第一个源字符，并为第二个源字符生成两个目标字符。根据对齐，中间结果变成了一个毫无意义的句子(“表表难过”，意思是“手表手表难过”)，导致错误的更正。

针对上述问题，提出了一种新的 NAR 模型 PhVEC(phonology-basedVariable-lengthErrorCorrection)。该模型融合了错误字符的音系特征，以实现变长预测。具体而言，PhVEC 包括一个检测网络和一个基于预训练语言模型的校正网络(Devlin et al.,2019;Sun et al.,2020)。检测网络预测每个 token 的正确性，修正网络生成的修正结果。我们没有使用长度预测器来预测每个源标记的生育能力，而是在每个检测到的错误字符之后，将语音标记(ChinesePinyin[1])作为占位符插入到源序列中。在预测期间，每个源 token(或字符 token 或拼音 token)

可以生成零或一个目标 token。在语音 token 的引导下，模型会生成发音相近的字符。如图 1(d)显示，拼音符号“b”产生的目标字符(“不”，bu)具有相同的辅音字母，“ao”产生的目标字符(“要”，yao)具有相同的元音字母。我们删除最后序列中重复出现的字符。

我们在具有不同程度 ASR 错误率的多个数据集上对方法进行了评估。实验结果表明，在相同的速度下，PhVEC 在标准测试集上比现有的基于 NAR 的 ASR 纠错方法获得了显著的性能提升(字错误率降低超过 10%)。即使与 AR 基线变压器和 BART(Lewis et al.,2020)模型相比，PhVEC 仍然可以有 8.55%和 2.39%的单词错误率降低，同时保持一个 6.2 倍加速。

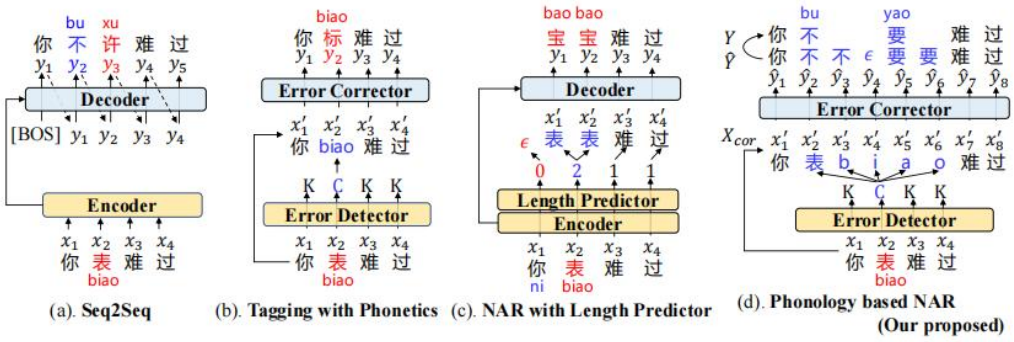



图 1:中文 ASR 错误校正示例。源语音“ni bu yao nan guo”(“不要难过”)被错误地识别为“ni biao nan guo”(“你的表难过了”)(a)序列到序列模型忽略了语音相似性，生成了流利但不正确的句子。(b)具有语音特征的序列标记模型正确地检测出了有错误的字符(“K”表示保持不变，“C”表示正确)，但由于它只支持等长度预测，因此无法生成以等长度作为输入的流畅更正。(c)带长度预测器的 NAR 模型错误地将源字符“ni”对应的目标标记数预测为 0，而源字符“ni”应该为 1。相应地，它删除了该字符并生成了不正确的结果。(d)提出的 PhVEC 算法首先检测出有问题的标记，然后利用它们的语音特征生成一个变长的中间结果，通过减少重复字符输出最终结果。

## 2 方法

中文 ASR 纠错可以形式化为以下任务。给定一个语音识别序列 X=$(x_1, x_2 \cdots x_n)$ 的 n 个汉字，目标是将其修正为另一个 m 个字符的序列 Y=$(y_1, y_2 \cdots y_m)$。需要注意的是，目标序列长度 m 并不一定等于源序列长度 n。从 X 到 Y 的转换过程中存在三种类型的 ASR 错误:替换、删除和插入。

如图 1(d)所示，我们提出的 ASR 误差校正模型由误差检测器和误差校正器组成，两者都执行 NAR 标记。错误检测器以 X 作为输入，预测每个令牌的正确性。纠错器则以源记号的组合和有问题记号的音韵特征作为输入进行修正。纠错时，扩展输入的每个标记都可以标记为汉字或空白标记“∈”，使模型能够支持变长纠错。由于不同的拼音字母可能会生成相同的字符，所以我们在最终结果中剔除了连续重复的字符。在本文中，我们使用汉语拼音作为语音特征。PhVEC 的学习是端到端进行的，错误器和纠错器共同优化。

## 2.1 错误检测网络

错误检测器的目标是检查一个字符 $x_i$ (1≤i≤n)是否正确。我们将这个任务建模为序列标记。我们构建一个序列二进制分类器，并使用类 1 和 0 分别标记有问题的字符和正确的字符。ground-truth 检测标签 C=$(c_1, c_2 \cdots c_n)$ 通过将 X 和 Y 与编辑距离匹配来预先计算，其中 $c_i \in \{0,1\}$。误差检测器的预测结果用序列 $C' = (c_1', c_2', \cdots c_n')$，我们用 $p_i$ 来表示 token $x_i$ 被预测到类 1 的概率，可以形式化如下:

$$p_i = p(c_i' = 1 \mid X) = soft\max(f_{dec}(E(e_w))) \quad (1)$$

在 $e_w = (e_{x1}, e_{x2}, \cdots e_{xn})$ 是 token X 的嵌入，E 是基于 transformer 的编码器，$f_{dec}$ 是将句子表示映射到二进制序列的全连接层。为了训练模型，我们采用以下交叉熵损失函数：

$$\Gamma_{dec} = -\frac{1}{n}\sum[c_i \ln p_i + (1-c_i)\ln(1-p_i)] \quad (2)$$

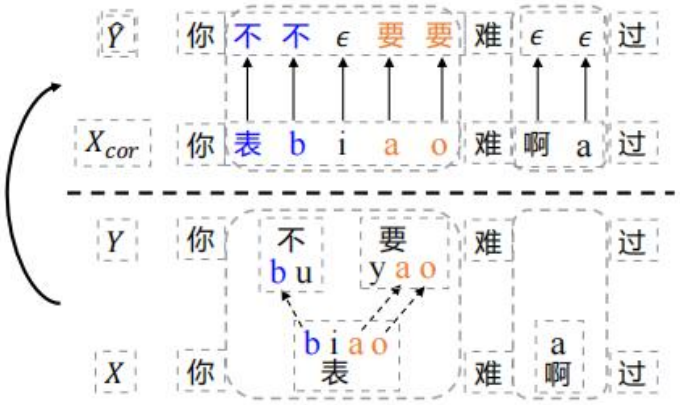



图2:生成的$X_{cor}$图解和Y来自训练样本(X,Y)。我们首先标注X中不正确字符的拼音和Y中对应字符的拼音(图的底部)。然后根据LCS算法(虚线箭头)对拼音标记进行对齐。$X_{cor}$是通过在X中错误字符后插入拼音tokens得到的。最后，根据对齐生成$\hat{Y}$。对于NULL对齐，例如“i”，我们对齐一个特殊的token“∈”。

## 2.2 纠错网络

在识别错误后，我们引入错误字符的拼音特征，并将其输入纠错网络。具体来说，我们使用*PyPinyin*[2]工具为每个错误的汉字生成拼音，并在有问题的拼音符号后插入拼音符号。最初的句子$X=(x_1,x_2,\cdots x_n)$则改写为$X_{cor}=(x_1^{'},x_2^{'},\cdots x_t^{'})$，其中$t\geq n$，同时t-n是拼音的token的数量。

鉴于$X_{cor}$，错误修正器运用序列标注来预测纠正结果$Y^{'}=(y_1^{'},y_2^{'},\cdots y_t^{'})$如下：

$$p(y_i^{'}=v_j\mid X_{cor})=soft\max(f_{crt}E(e_{cor})) \quad (3)$$

其中$p(y_i^{'}=v_j\mid X_{cor})$是$x_i$被正确纠正为$v_j$（词汇表$v$中的索引为j的token）的概率。$e_{cor}$是$X_{cor}$的token嵌入，$f_{crt}$是一个将token隐藏层映射到器长度为$|v|$的预测logit向量的全连接层。注意，token嵌入的参数，编码器E和校正网络$f_{crt}$由预训练的语言模型初始化(Devlin et al.,2019;Sun et al.,2020)。我们在纠错器和纠错器中共享预训练模型参数，对共享空间中的汉字和拼音token进行编码，这不仅减小了模型的大小，也使模型产生了更好的语义表示。

纠错器的学习目标是纠正 $X_{\text{cor}}$ 到最好的修正 Y。然而，这并不是一项简单的任务，因为 $X_{\text{cor}}$ 的长度和 Y 的长度可能不一样。因此，我们将 Y 重写为 $\hat{Y}=(\hat{y}_1,\hat{y}_2,\cdots\hat{y}_t)$，其长度与 $X_{\text{cor}}$ 相同。具体地说，$\hat{Y}$ 是由 Y 构造而成，并根据发音对齐重复插入一些符号。我们将 X 的错误 token 的拼音与他们相应的正确的 Y 进行比较，并根据最长公共子字符串(LCS)算法进行对齐，如图 2 底部虚线箭头所示。然后我们根据对齐重写 Y 到 $\hat{\text{y}}$。误差校正器的损失可以定义如下:

$$\Gamma_{\text{cor}}=-\sum_{i=1}^{\text{t}}\log(p(y_i^{'}=\hat{y}_i\mid X_{cor})) \quad (4)$$

我们提出的拼音对齐为模型学习拼音和汉字之间的相关性提供了明确的线索。作为一种替代方法，我们还可以通过连接时态分类(Connectionist Temporal Classification,Graves et al.,2006)来学习对齐(它广泛用于变长对齐，如 ASR(Audhkhasi et al.,2019)、手写识别(Bluche et al.,2014)等。但我们没有在模型中使用 CTC，因为我们假设根据语音特征对齐可以可靠地进行 ASR 纠错，与学习全局对齐相比，这会降低学习的难度。我们在实验中证实了我们的假设。

## 2.3 联合学习与推断

对于每个训练样本(X,Y)，我们首先构造它的黄金检测结果 C 和修正目标 Y，然后联合优化两个模块，如下所示:

$$\Gamma=\lambda\Gamma_{\text{dec}}+(1-\lambda)\cdot\Gamma_{cor} \quad (5)$$

其中 $\lambda$ 为权衡参数。

在推理阶段，我们首先检测出有问题的字符，并将其拼音插入到原句子中，然后通过最大化每个 i 的 $\text{p}(y_i^{'}\mid X_{cor})$ 的概率来预测修正序列 $Y^{'}$，最终的修正结果是通过去除空白符号产生的∈并合并来自纠错器预测的相邻重复字符，如图 1(d)顶部所示。注意，这样的后处理只在 $X_{cor}$ 相对于 X 进行修改，而由错误检测器预测为正确的字符在最终预测中保持不变。

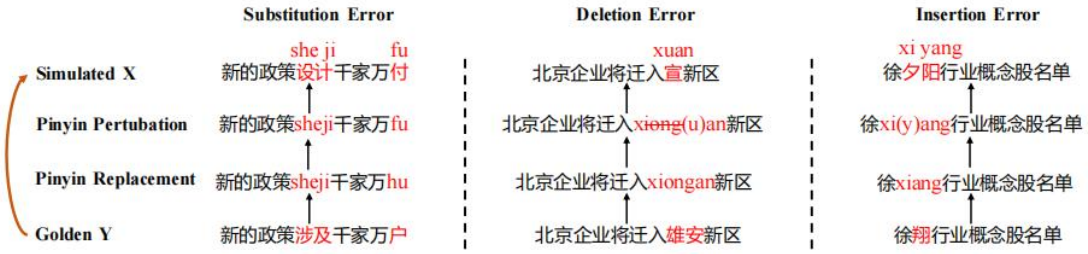



图 3:生成训练样本的过程。从下到上用红色标记的字符分别代表正确的汉字、正确汉字对应的拼音、模拟的拼音扰动、噪声拼音对应的模拟错误字符。

## 3 训练数据生成

我们遵循纠错中的常见做法(Zhang et al.,2020;Takahashi et al.,2020;Leng et al.,2021)，利用模拟的 ASR 误差合成训练语料。该方法通过将干净句子中的汉字或单词替换成有问题的汉字或单词来生成模拟的 ASR 结果。具体来说，我们分三步生成噪声文本:(i)采样一些候选字符，并将其替换为拼音。(ii)在原始拼音上添加噪声并生成新的有效拼音(iii)基于更新后的拼音生成新的汉字或单词。

在第一步中，从包含容易被错误识别的单词的混淆集(Wang et al.,2019)中获得候选单词。在第二步中，我们设计了三种拼音扰动策略来模拟替换错误、删除错误和插入错误，如图 3 所示。注意，在生成删除和插入错误时，可能会插入一些随机字母来构建有效的拼音序列。最后，用候选词替换噪声拼音，选取 n-gram 语言模型排序最高的句子作为最终的模拟 ASR 结果。

## 4 实验

在两个中文 ASR 纠错数据集上进行实验。我们使用单词错误率(WER)和单词错误率(WERR)来评估我们的纠错性能。

### 4.1 数据设置

我们首先在 AISHELL-1 上训练 ASR 模型(Bu et al.,2017)。训练集包含 150 小时的普通话演讲，以及相应的转录本，主要包括财经、科技、体育等方面的新闻。然后，训练好的 ASR 模型转录语音，生成配对数据供评估，如表 1 所示。AISHELL-1 开发集和

测试集包含 15 小时的语音，对应 21K 对句子。MAGICDATA3 包含 43 小时的普通话演讲，包括互动问答、日常指导等。

由 300 万条网络抓取的句子以及带有模拟错误的噪声句子组成 ASR 纠错训练集。对于只支持定长校正的方法，只产生具有相似发音替换错误的伪 ASR 输入。对于支持变长校正的方法的训练数据，我们从定长训练数据中抽取 270 万个样本，并为剩下的 0.3M 句生成模拟插入和删除错误。

## 4.2 训练细节

与之前的工作(Leng et al.,2021 年)一致，我们使用 ESPnet(Watanabe et al.,2018)工具包在 AISHELL-1 训练集上训练 ASR 模型。Conformer architecture(Gulati et al.,2020)和 SpecAugment(Park et al.,2019)也被用于提高 ASR 性能。对于 ASR 纠错网络，我们采用”chinese-bert-wwm”(Cui et al.,2019)作为预训练模型来初始化我们的 PhVEC 编码器和其他基于预训练的方法。我们将学习率设置为$5e^{-5}$，dropout 设置为 0.1。联合学习中的损失平衡参数 λ 设置为 0.5,并利用 AdamW(Loshchilov and Hutter,2019)作为优化器。更多细节可以在附录 A 中找到。

| Dataset | AISHELL-1 | MAGICDATA |
|---|---|---|
| Dev | 14,329 | 6,756 |
| Test | 7,176 | 15,131 |
| Avg Length | 14.42 | 9.78 |
| Type | News | Command |

表 1:实验测试数据集的统计结果。

## 4.3 基线

我们将我们的 PhVEC 与自回归变压器(Vaswani et al.,2017),BART 进行了比较(Lewis et al.,2020)同时最近最先进的非自回归方法如

下:Levenshtein Transformer(LevT)(Gu et al.,2019)通过在 NAR 框架下迭代执行删除、插入和替换来支持变长校正。FastCorrect(Leng et al.,2021)使用长度预测器

实现变长校正，该预测器估计每个源 token 应该转换为目标 token 的数量，然后重复/丢弃源 token，以生成变长序列作为错误校正的输入。MLM-phonetics(Zhang et al.,2021a)通过预训练集成语音特征的语言模型来实现固定长度的错误校正。BERT 直接微调标准掩码语言模型以生成固定长度的修正。BERT+CTC 算法在 BERT 算法的基础上构建了一个 CTC 层来实现变长纠错，该层对每个源标记进行 2 次上采样，在保留 CTC 的情况下自动学习源目标比对。

## 4.4 总体结果

两个基准数据集的比较结果如表 2 所示。我们观察到:

•我们提出的 PhVEC 在评估数据集上的性能优于所有其他方法。对于 ASR 性能不同的 4 个数据集，算法均获得了约 20%的 WERR。PhVEC 大大超过了现有的两种支持变长校正的方法 LevT 和 FastCorrect，成为第一个超过自回归 Transformer 和 BART 模型的方法，WERR 平均提高了 8.55%和 2.39%。

•PhVEC 和 MLM-phonetics 都将语音特征引入模型，但 PhVEC 性能更好。我们将这种改进归结于两个方面:一是它对变长错误的有效解决，二是拼音分裂策略的有效使用。与 Zhang et al(2021a)不同，我们在字母粒度上使用拼音特征，而不是将每个拼音视为一个标记。这有利于灵活插入，并增强了相似发音的字符与其对应的语音特征之间的相关性。

•训练前显著提高矫正成绩。对于自回归模型，预训练的 BART 模型比传统的 Transformer 模型具有更低的 WER。同时，优化后的 BERT 性能与 Transformer 相当，这表明强语言建模将极大地促进 NAR 方法的错误校正。在基于预训练的方法中，PhVEC 仍然表现最好，表明了利用拼音标记进行变长校正的有效性。

•添加 CTC 层并没有给纠错带来明显的优势。特别地，BERT+CTC 不如 BERT，在固定长度校正样本中容易产生不正确的对准。这可能是因为，对于 ASR 误差校正，对准源和目标的大部分词之间是确定的。BERT+CTC 对每个源标记进行 2 次上采样，并指示模型学习对齐，实际上增加了学习的难度。值得注意的是，BERT+CTC 在变长纠错样本上的性能优于 BERT，稍后将在表 3 中介绍。

•PhVEC 加速到变压器的 6.2 倍，并进一步降低 8-10%的 WER。与 BART 模型相比，PhVEC 的 WER 降低了 2-3%，推理速度提高了 7.5 倍，证明了 PhVEC 的高效性和有效性。另

外，在此强调一下，原来 FastCorrect 分别使用 NVIDIAV100 和 P40GPU 测试了模型延迟。为了保证推理结果的一致性，在 NVIDIAV100GPU 上实现了所有的基线方法，并测试了推理速度。

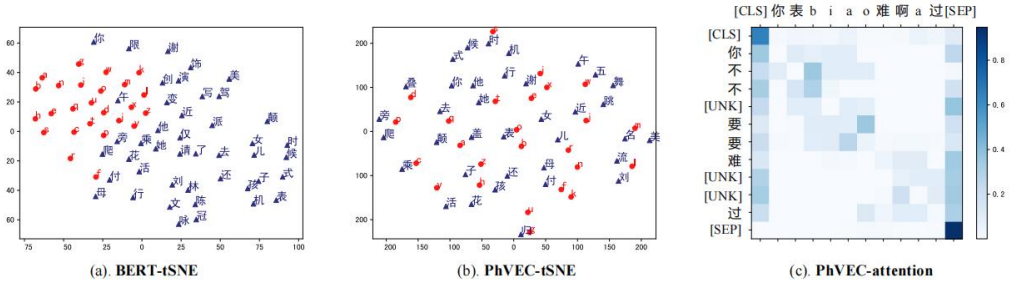
值得注意的是，我们的方法 PhVEC 的训练数据包括 3M 对句子，这与 Transformer、BART、LevT、BERT 和 BERT+CTC 的训练数据是一致的，但 FastCorrect 和 MLM-phonetics 分别额外使用了 400M 和 300M 对句子进行预训练。

| Method | AISHELL-1 | | MAGICDATA | | Latency(ms/sent) |
|---|---|---|---|---|---|
| | Dev | TEST | Dev | TEST | TEST |
| | WER↓ (WERR↑) | WER↓ (WERR↑) | WER↓ (WERR↑) | WER↓ (WERR↑) | GPU |
| No correction | 4.46 (-) | 4.83 (-) | 13.82 (-) | 13.51 (-) | - |
| Transformer | 3.80 (14.80%) | 4.08 (15.53%) | 12.09 (12.52%) | 12.19 (9.77%) | 149.5 (1×) |
| BART | 3.62 (18.83%) | 3.81 (21.12%) | 11.36 (17.80%) | 11.33 (16.14%) | 180.9 (0.8×) |
| LevT (Max iter=1) | 4.37 (2.02%) | 4.73 (2.07%) | 13.81 (0.07%) | 13.91 (-2.96%) | 54.0 (2.8×) |
| FastCorrect | 3.89 (12.78%) | 4.16 (13.87%) | - | - | 21.2 (7.1×) |
| BERT | 3.71 (16.82%) | 3.98 (17.60%) | 11.67 (15.56%) | 11.79 (12.73%) | 14.1 (10.6×) |
| BERT + CTC | 3.78 (15.25%) | 4.01 (16.98%) | 12.13 (12.23%) | 12.11 (10.36%) | 14.5 (10.3×) |
| MLM-phonetics | 3.64 (18.39%) | 3.90 (19.25%) | 11.59 (16.14%) | 11.68 (13.55%) | 23.7 (6.3×) |
| PhVEC (Ours) | **3.52 (21.08%)** | **3.62 (25.05%)** | **11.19 (19.03%)** | **11.04 (18.28%)** | 24.1 (6.2×) |

表 2:我们的方法与其他基线在开发集和测试集上的性能比较。WER 为字错误率(%)，WERR 为字错误率减少率。在 NVIDIAV100GPU 上测试了修正模型的推理速度，并将测试批大小设置为 1 句，以匹配在线服务环境。

## 4.5 分析

我们进一步分析了我们的模型在变长数据集上的性能，并进行了消融研究剖析了影响我们方法有效性的因素。




(a). **BERT-tSNE** (b). **PhVEC-tSNE** (c). **PhVEC-attention**

图 4:(a)和(b)为拼音字母和汉字嵌入的 t-SNE 可视化，分别对应 BERT 模型和 PhVEC 模型。(c)为3rd 的自我关注误差校正器的编码器层。横轴表示 $X_{cor}$，错误校正器的输入，垂直标记是模型输出，其中[UNK]是空白标记∈。

### 4.5.1 变长场景

为了评估模型处理变长误差的有效性，从 AISHELL-1 和 MAGICDATA 的测试集中抽取包含插入和删除错误的样本，并在该子集上评估上述方法。如表 3 所示，与其他方法相比，PhVEC 表现出明显的优势，在两个数据集上都达到了 20%以上的 WERR。精细调优 BERT 和 MLM-phonetics 的性能都很弱，因为它们的生成具有固有的固定长度限制。CTC 层实现了 BERT 的变长预测，使其性能得到了显著提升，但与 PhVEC 相比仍有较大差距。LevT 产生的校正不稳定，在 MAGICDATA 子集中，WERR 甚至降低了 8.1%。这可能是因为它的长度预测结果不准确。输入长度与黄金校正长度的平均差仅为 1.08，而左心室预测长度与黄金校正长度的平均差为 1.42。这说明左心室的预测会产生比原始 ASR 输入更多的长度差异，导致其性能下降。

| Strategy | Example | AISHELL-1 | MAGICDATA |
|---|---|---|---|
| No correction | [表] | 4.83 (-) | 13.51 (-) |
| *OneToken* | [表][biao] | 4.08 (15.5%) | 12.11 (10.4%) |
| *Initial&Final* | [表][b][iao] | 3.91 (19.0%) | 11.68 (13.5%) |
| *Letters-only* | [b][i][a][o] | 3.85 (20.3%) | 11.59 (14.2%) |
| *Letters* (Ours) | [表][b][i][a][o] | **3.62 (25.1%)** | **11.04 (18.3%)** |

表 4:在 AISHELL-1 和 MAGIC-数据测试集上评估的不同语音特征的 WER↓(WERR↑)。

### 4.5.2 利用音韵特征的不同方式

在我们的模型中，我们将每个汉字的拼音一个字母一个字母地分割为 token，并在每个检测到的问题字符之后添加它们，例如，将检测到的错误字符“[表]”(biao)重写为中间变长结果中的 5 个 token[表][b][i][a][o]。在此，我们探索了利用不同策略

来利用拼音的影响:1)OneToken:像 Zhangetal.(2021a);将每个字符的拼音作为一个 token;2)Initial&Final:根据汉语的音韵部分，将每个拼音分为一个开头的([b])和一个最后的([iao]);3)Letters-only:将原有问题的字从重写的句子中去掉，用其拼音代替。前两种设置侧重于不同的拼音颗粒，第三个想检查被检测为错误的原始字符是否对纠错有用。

表 4 显示:(1)与字母相比，OneToken 和 Initial&Final 都会降低性能。这是因为两种方法所使用的语音特征的粒度比字母的粒度更粗，降低了每个语音特征与更多汉字对齐的概率。(2)Letters-only 的性能低于 Letters，证明了在 $X_{\text{cor}}$ 中保留原始字符的有效性。这可能是因为，一些错误的检测结果将“C”标记为一些正确的单词。在添加拼音标记时删除这些原始单词，使得模型几乎不可能恢复正确的单词。这种情况总是发生在一些命名实体中的生僻词上。

### 4.5.3 字符嵌入和对齐

为了定性地检查 PhVEC 是否学习了有意义的表示，我们深入学习的字符嵌入和编码器进行可视化。我们首先调查模型是否学习了拼音字母和汉字之间的关系。具体来说，我们通过应用 t-SNE 算法(van der Maaten and Hinton,2008)，将学习到的拼音字母和汉字嵌入在二维空间中可视化。从图 4b 可以看出，经过 PhVEC 训练后，汉字与其对应的拼音字母的嵌入更加接近。然后我们在纠错器中可视化编码器的自注意。图 4c 显示输出中被纠正的单词非常关注其对应的拼音令牌，例如“不(bu)”与“b”高度对齐，“要(yao)”则非常关注“a”和“o”，这表明语音特征为纠错提供了明显的提示。

## 5 相关工作

### 5.1 中文拼写误差校正(CSC)

作为与 ASR 纠错密切相关的领域，近年来得到了广泛的探索。Zhang et al.(2020)提出了一种软屏蔽 BERT 模型，该模型首先预测每个字符的错误概率，然后利用这些概率进行软屏蔽词嵌入进行修正。作为软蒙面 BERT 的补救措施，Zhang et al.(2021a)将语音知识融入了预训练，并提出在纠错中融合语音特征。Cheng et al.(2020)在

BERT 的基础上构建了图卷积网络，该网络反映了汉语符号之间的语音相似性。然而，这些方法被设计成产生与输入长度相同的修正，但无法处理包括替换、删除和插入错误在内的变长修正。

## 5.2 自回归(AR)误差校正

为了校正变长误差，大量序列到序列 AR 模型被提出。Zhang et al.(2019)使用了一种基于变压器的中国 ASR 误差校正模型。Wang et al.(2019)将复制机制集成到序列到序列框架中，直接从准备好的错误单词混淆集复制更正。随着预训练的普及，Zhao et al.(2021)使用预训练的 BART(Lewis et al.,2020)来初始化纠错网络。虽然这些 AR 模型能够处理 ASR 中的各种类型的错误，但它们不能满足在线服务的延迟需求，特别是对于一些实时场景，如同步翻译(Zhang et al.,2021b)。

## 5.3 非自回归(NAR)误差校正

与 AR 模型相比，NAR 模型通过并行产生目标序列中的所有标记，从而获得了更快的生成速度，这在机器翻译中得到了广泛的探索(Gu et al.,2019;Xu 和 Carpuat,2021)、ASR(Fanetal.,2021)、TTS(Ren et al,2019)等。最近，一些研究(Gu et al.,2019;Leng et al.,2021)提出将 NAR 模型应用于基于长度预测的变长 ASR 纠错，该模型首先估计目标纠错的长度，然后根据估计的长度，通过丢弃或重复一些标记来重写输入句子，最后对重写的句子进行序列标记来实现纠错。然而，即使对人类来说，也很难直接预测错误句子的目标长度，而长度预测的准确性与纠错的性能密切相关。

# 6 结论

本文提出了一种基于音素训练的非自回归中文 ASR 纠错网络。我们的方法首先检测出有问题的字符，然后加入它们的语音特征来调整输入长度，从而生成一个变长序列进行纠错。语音特征使我们的模型能够产生相似的发音纠正，并支持非自回归模式下的变长纠正。实验表明，我们的方法在保持 6.2 倍加速的情况下优于自回归方法。作为未来的工作，我们计划将 PhVEC 扩展到其他语言，并使用相应的语音标记来纠正发音引起的变长错误。

## 确认

国家重点研发计划项目(NO.2017YFC0820700)和国家自然科学基金项目(No.61902394)资助了本研究。感谢所有作者的贡献，感谢所有匿名审稿人的建设性意见。

### 参考文献

C. Anantaram, Amit Sangroya, Mrinal Rawat, and Aishwarya Chhabra. 2018. Repairing ASR output by artifificial development and ontology based learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artifificial Intelligence, IJCAI 2018*, pages 5799–5801. ijcai.org.

Kartik Audhkhasi, George Saon, Zoltán Tüske, Brian Kingsbury, and Michael Picheny. 2019. Forget a bit to learn better: Soft forgetting for ctc-based au
tomatic speech recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 2618–2622. ISCA.

Théodore Bluche, Hermann Ney, and Christopher Kermorvant. 2014. A comparison of sequence-trained deep neural networks and recurrent neural networks optical modeling for handwriting recognition. In *Statistical Language and Speech Processing - Second International Conference, SLSP 2014*, volume 8791 of *Lecture Notes in Computer Science*, pages 199–210. Springer.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. In *Oriental COCOSDA 2017*, page Submitted.

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. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 871–881. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pretraining with whole word masking for chinese BERT. *CoRR*, abs/1906.08101.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the*

*Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171 – 4186. Association for Computational Linguistics.

Ruchao Fan, Wei Chu, Peng Chang, Jing Xiao, and Abeer Alwan. 2021. An improved single step non-autoregressive transformer for automatic speech recognition. *CoRR*, abs/2106.09885.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006)*, volume 148 of *ACM International Conference Proceeding Series*, pages 369 – 376. ACM.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018*. OpenReview.net.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 11179 – 11189.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association*, pages 5036 – 5040. ISCA.

Jinxi Guo, Tara N. Sainath, and Ron J. Weiss. 2019. A spelling correction model for end-to-end speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 5651 – 5655. IEEE.

Oleksii Hrinchuk, Mariya Popova, and Boris Ginsburg. 2020. Correction of automatic speech recognition with transformer sequence-to-sequence model. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020*, pages 7074 – 7078. IEEE.

Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiang-Yang Li, Ed Lin, and Tie-Yan Liu. 2021. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition. In *35th Conference on Neural Information Processing Systems, NeurIPS 2021.*

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, ACL 2020*, pages 7871–7880.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net.

Anirudh Mani, Shruti Palaskar, Nimshi Venkat Meripo, Sandeep Konam, and Florian Metze. 2020. ASR error correction and domain adaptation using machine translation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 6344 –
6348. IEEE.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association*, pages 2613–2617. ISCA.

Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. 2020. A study of non-autoregressive model for sequence generation. *arXiv preprint arXiv:2004.10454*.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 3165–3174.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artifificial Intelligence*, volume 34, pages 8968–8975.

Yujin Takahashi, Satoru Katsumata, and Mamoru Komachi. 2020. Grammatical error correction using pseudo learner corpus considering learner’s error tendency. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, ACL 2020, Online, July 5-10, 2020*, pages 27–32. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 5998– 6008.

Dingmin Wang, Yi Tay, and Li Zhong. 2019. Confusionset-guided pointer networks for chinese spelling check. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 5780–5785. Association for Computational Linguistics.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. Espnet: End-to-end speech processing toolkit. In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*, pages 2207–2211. ISCA.

Weijia Xu and Marine Carpuat. 2021. EDITOR: an edit-based transformer with repositioning for neural machine translation with soft lexical constraints. *Trans. Assoc. Comput. Linguistics*, 9:311–328.

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. 2021a. Correcting chinese spelling errors with phonetic pre-training. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 2250–2261. Association for Computational Linguistics.

Ruiqing Zhang, Xiyang Wang, Chuanqiang Zhang, Zhongjun He, Hua Wu, Zhi Li, Haifeng Wang, Ying Chen, and Qinfei Li. 2021b. BSTC: A large-scale Chinese-English speech translation dataset. In *Proceedings of the Second Workshop on Automatic Si
multaneous Translation*.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 882–890. Association for Computational Linguistics.

Shiliang Zhang, Ming Lei, and Zhijie Yan. 2019. Investigation of transformer based spelling correction model for ctc-based end-to-end mandarin speech

recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 2180 - 2184. ISCA.

Yun Zhao, Xuerui Yang, Jinchao Wang, Yongyu Gao, Chao Yan, and Yuanfu Zhou. 2021. BART based semantic correction for mandarin automatic speech recognition system. *CoRR*, abs/2104.05507.

# A 实验细节

## A.1ASR 模型的结构和参数

ASR 模型是一个端到端的编码器-注意力-解码器模型，包含一个 12 层 conformer 编码器和一个 6 层 conformer 解码器，在解码器输出上使用交叉熵损失和编码器输出上使用辅助 CTC 损失进行训练。对于 ASR 模型的超参数，我们采用波束搜索译码，波束大小为 10，保形核大小为 15,ctc 权重为 0.6,lm 权重为 0.3。

## A.2 平衡检测和校正的目标

我们探讨了在微调中平衡两个目标的加权策略的影响。表 5 给出了 PhVEC 在不同超参数 λ 值下的结果。具体来说， λ 值越大意味着误差检测的权重越高， λ 为 0.5 时 F1 得分最高。

| λ | AISHELL-1 | | MAGICDATA | |
|---|---|---|---|---|
| | Dev(WER↓) | TEST(WER↓) | Dev(WER↓) | TEST(WER↓) |
| 0.2 | 3.61 | 3.75 | 11.42 | 11.37 |
| 0.5 | 3.52 | 3.62 | 11.19 | 11.04 |
| 0.8 | 3.74 | 3.83 | 11.55 | 11.47 |

表 5:λ 不同值的影响。

# 文献综述

# 面向语音识别文本的纠错系统研究综述

## 摘要

随着自动语音识别（ASR）技术的不断普及，实现一个面向语音识别文本的纠错系统的需求越来越迫切。在 ASR 的过程中，由于说话人的背景环境嘈杂或者说话人的吞咽、拖拽、口音、方言等现象导致语言转换后的文本难以理解，文本中存在较多的错误。同时由于中文和英文的区别，相关领域一些对英文的处理方法无法直接运用到中文中。本文借鉴相关领域的研究成果，对中文自动语音识别(ASR)的研究现状及方法进行了介绍评估。

## 1 引言

相较于传统文本纠错，语音文本有着自己的特色，纠错方法也和普通文本的方法有些不同。在语音识别系统中，一般来说，文本的错误类型.有四种。包括冗余词、漏词、选词错误和混合错误。如将“pian”识别为“ping”和“an”，形成冗余词。由于口音或方言将“na”识别为“nei”,从而形成“内”等字。由于说话人语速过快会将第一个汉字和第二个汉字拼在一起，如“xi an”识别为“xian”。最后一种是前面的总和，将“学不可以已”，翻译成“雪不可以”，“学”和“雪”同音以及将“yiyi”识别为“yi”[1]。因此如何处纠正 ASR 后的文本有着重要的意义。

现今对自动语音识别(ASR)出的文本的处理模型主要分为自回归模型和非自回归模型。两者各有自己的特点。

自回归模型：给定一个序列，使用深度神经网络对序列里面的每一个变量进行条件概率建模，每个变量的条件概率依赖于当前变量在序列里的所有前置变量。自回归符合人们单向阅读的直觉和思维方式。采用迭代循环的方式，通过编码器对源语句进行编码后输至解码器，然后解码器根据源语句和上一步预测的目标端语言单词来预测下一个单词，一步步的输出最后的句子。模型可基于注意力(Ghazvininejad[2]等人)、

BLSTM(Mani[3]等人)等。

非自回归模型：相较于自回归模型，非自回归模型在长句子处理的速度上有着很大的优势，相比于一个一个递归循环解码，非自回归模型一次性并行生成句子中所有词语。但是非自回归模型模型由于没有很好的考虑词语间的条件依赖，正确率要稍微低一些。

针对以上情况，现在工作提供了很多解决方法[4]：

数据层面：利用知识蒸馏[5]来生成数据、设计数据学习算法等。利用预训练 NAR 模 型，基于知识蒸馏的方法将训练集中的源语句进行翻译，并将源语句和翻译结果作为 NAR 模型的训练集。这种方式可以减少训练数据的多 样性，减轻 NAR 模型的训练难度。

模型层面：设计迭代式模型、基于隐变量的模型以及增强解码器模型结构等。其中，迭代式模型将原始一次解码的 NAR 模型扩展成为多次迭代解码的模型，这样在进行每轮迭代时，上一轮迭代的结果可以作为目标语言端的依赖信息，将一次解码的难度分摊到多次迭代中，从而提升 NAR 模型的效果。

损失函数层面：主要针对传统交叉熵损失函数的问题，提出一系列改进方法，包括基于 CTC[6]、n-gram、以及引入顺序信息的损失函数。其中，由于 n-gram 的方法针对传统的交叉熵损失函数只能提供单词级别的监督信息而无法 提供全局信息，研究员们提出了优化预测和目标之间 Bag of N-gram 差异的损失函数，以补充交叉熵损失函数中缺失的全局信息，以更好地对 NAR 模型进行优化。

解码算法层面：对 NAR 模型的长度预测模块进行改进，以及对传统解码算法的改进。由于 NAR 模型无法像 AR 模型一样隐式地在解码过程中决定目标语句的长度，因此需要在解码过程开始前就对目标语句的长度进行显式预测。这个步骤十分重要，因为目标语句的长度是否匹配直接影响模型最终的翻译效果。因此，类似自回归解码中的 Beam Search[7]，有模型提出了提升长度预测准确率的方法，如多个长度并行解码等。

利用预训练模型：利用自回归翻译模型和利用单语大规模预训练语言模型的方法。其中，由于 NAR 模型和 AR 模型结构相似，并且 AR 模型的翻译准确度更高，因此很多方法提出利用预训练的 AR 模型来额外监督 NAR 模型的训练，包括在隐变量层

面引入额外监督信息，和基于课程学习的迁移学习方法等。

# 2 研究现状

构建自动语音识别文本的纠错系统通常由两种方法：

第一种是多模型 Pipeline 把纠错任务分为多个步骤多个模型，每个模型解决具体的特定问题，然后通过 Pipeline 的方式串联，得到最后结果；这种方式的优点在于方便分析问题，能够针对各个问题进行重点突破，但是也存在问题。

第二种系统则采用端到端的学习、联合优化，单个模型就能完成纠错任务。

## 2.1 多模型 Pipeline 方法

主要分为三部分：错误检测、候选召回、候选排序。

### 2.1.1 错误检测：

找到句子可能存在的错误，定位到错误词的位置。

1)基于规则：

拼音匹配检测: 抽取 n-gram[8]级别的拼音，构建实体词与拼音映射词表，将抽取的拼音映射到实体词，比较和原词是否一致。

拼音编辑距离检测: 同上，但是拼音映射的实体词并不一定要完全同拼音，一定编辑距离之内即可。

单双向 2-gram 检测: 统计大规模语料中 2-gram 的频率，检测适合句子中的 2-gram 小于一定频次即识别为错误。

困惑集：在 SIGHAN7 Bake-off 2013 中文拼写检查任务中，组织者提供了六种混淆集：4 组语音相似的字符（音近字）和 2 组视觉上相似的字符（形近字）

2)语言模型：

使用 LM 预测下一个字（或上一个字），如果真实字不在预测的 top-k 内，即识别为错误。或者使用 BERT 的遮蔽语言模型，将当前词用 mask 符号代替，然后做预

测，若真实字不在 top-k 内即为错。

3)序列标注或者 PLM：

在 PTM 流行之前，主流做法是 Bi-LSTM+CRF 或者 Transformer+CRF 预测每个字是否有错，现在也可以用 Bert 等预训练模型。

### 2.1.2 候选召回：

候选召回指的是，识别出具体的错误点之后，需要进行错误纠正，为了达到更好的效果以及性能，需要结合历史错误行为，以及音形等特征召回纠错候选。

主要可分为两部分工作：离线的候选挖掘，在线的候选预排序。

离线候选挖掘利用大规模多来源的错误对齐语料，通过对其模型，得到不同粒度的错误混淆矩阵。

在线候选预排序主要是针对当前的错误点，对离线召回的大量纠错候选，结合语言模型以及错误混淆矩阵的特征，控制进入纠错排序阶段的候选集数量与质量。

现在主流的召回纠错候选方案都是结合规则与深度学习方法，同时控制一部分词进入候选集。

### 2.1.3 候选排序：

主要有粗排序和精排序，候选召回模块得到的候选词数量庞大，逐一通过复杂模型计算替换概率将引入较大的时间损耗，因此在精排序前需要进行一定粗排序，从而通过简单的算法来过滤掉部分明显错误的答案。

## 2.2 端到端

构建中文纠错系统的第二种方法是构建端到端的系统。机器翻译方法是目前最为热门的方法，简单的说，它将从原始“错误句子”到修正后“正确句子”的处理过程，看作是一个机器翻译的问题，即将错误的句子翻译成正确的句子。

此方案在 PTM 流行之前是纠错竞赛的标配，一般会结合之前三段式的方案组合

使用。

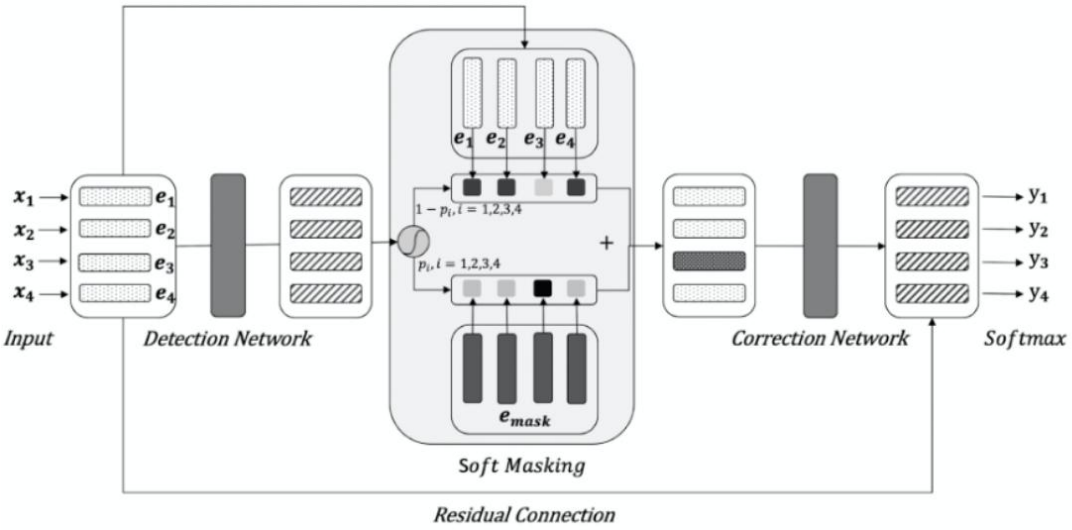
### 2.2.1 Soft-Masked Bert

由于 Bert 无法判断文本中的词是否需要纠错，缺乏检测能力。为此，提出了一个包含检测网络跟纠正网络的中文纠错方法[9]。

文本首先是经过检测网络，然后再经过纠错网络。其中检测网络是的双向 GRU+全连接层做一个二分类任务，计算原文本每个位置是否错误的概率。每个位置有错别字的概率为 p，没有错别字的概率是 1-p，。纠正网络采用的是预训练模型 Bert，但是在嵌入层的地方有所不同，每个位置的嵌入是由原文本中对应位置的字的词嵌入跟[MASK]的词嵌入的加权和得到的，这里的[MASK]的权重等于检测网络预测的当前位置是错别字的概率。

具体如图(a)所示，所以如果检测网络判断当前位置是错别字的概率较高，那么在纠正网络中该位置的词嵌入中[MASK]的权重就更高，反之，如果检测网络判断当前位置是错别字的概率很低，那么在纠正网络中该位置的词嵌入中[MASK]的权重就更低。利用 Bert 获得每个位置的表征后，将 Bert 最后一层的输出加上原文本中对应位置的词嵌入作为每个时刻最终的表征，通过全连接层+Softmax 去预测每个位置的字，最终选择预测概率最大的字作为当前结果的输出。

训练过程中联合训练检测网络跟纠正网络的，模型的目标包括两部分，一个是检测网络的对数似然函数，另一个是纠正网络的对数似然函数，通过加权求和联合这两部分，使得加权和的负数尽可能小，从而同时优化这两个网络的参数




图(a)

## 2.2.2 MLM-phonetics

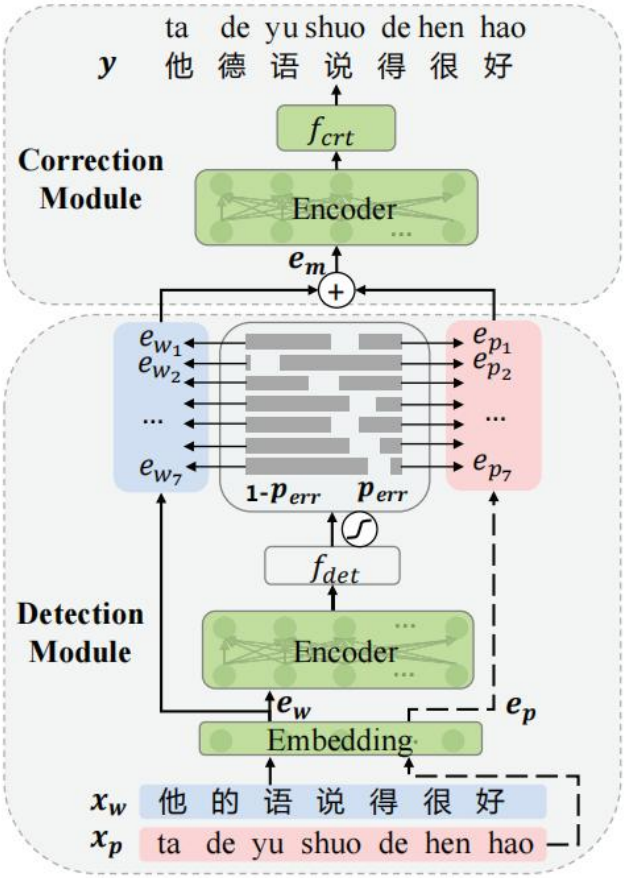
此模型[10]是在 Soft-Masked BERT 做出的改进。其注意到了自动语音识别文本中错误很多来自于同音字，所以将拼音信息融入到了模型中。模型如图(b)。

Soft-Masked BERT 的词嵌入由原文本中各个位置本身的词嵌入和[MASK]的词嵌入组成，而 MLM-phonetics 则是将相应[MASK]的词嵌入替换为相应位置对应的拼音序列的嵌入。

MLM-phonetics 在纠正网络的目标函数中加入了检测网络的预测结果作为一个权重项。

MLM-phonetics 的检测网络采用了预训练模型 Bert。

为了更加适配中文纠错任务的场景，MLM-phonetics 的 Bert 的 MLM[11]任务中预测的字都是根据汉字常见的错误选取的，要不在字形上有相似之处，要不在发音上有相似之处。




图(b)

### 2.2.3 GAD

由于 BERT 的每一层 Transformer，对于某一个 token 的 representation，都是对所有的上一层的 token representation 进行加权求和，由于存在错误的 token，导致这部分的噪声信息也被进行了加权，因此会对 CSC 模型产生影响。因此，作者提出 GAD 模型[12]，即对每一层 Transformer，计算 attention 时，还要考虑错误 token 对应的 confusion set 中高置信度的其他 token。

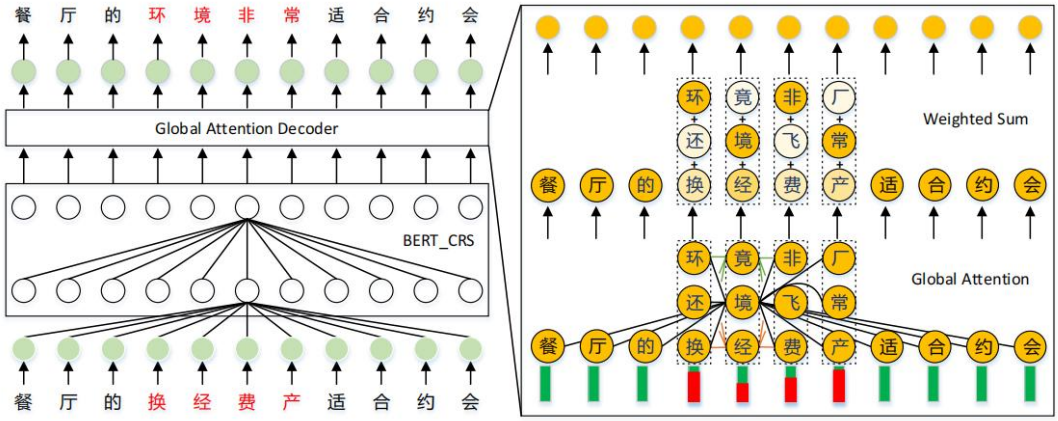
主要两个步骤：

1 给定一个待纠错的中文字符序列， 首先通过 BERT_CRS 模块（下文介绍）获得每个字符的表征向量，其次通过 decision network 将其映射为 0-1 序列，其中 0 表示当前的字符是正确的，1 表示当前的字符是错误的；

2 在 BERT_CRS 之上，添加 Global Attention Decoder（GAD）模型，其融入额外

的候选字符，以避免局部上下文的错误信息；

模型框架如图(c)

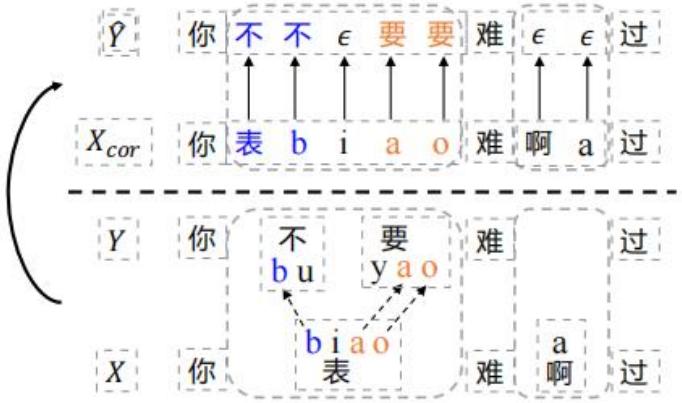



图(c)

## 2.2.4 PhVEC[13]

相比于其他模型(一些固定长度预测[14][15]，和不考虑音素的变长预测[16])，此模型注重的是如何再考虑因素关系的同时实现变长预测。本文将检测到的错误插入拼音token。根据发音对齐重复的插入一些符号。本文没有使用 CTC，在实验中验证了通过语音特征对齐可以更好的进行 ASR 纠错。

数据处理策略如图(d)




图(d)

# 3 数据集

Magicdata-RAMC[17]：

包含 351 组多轮普通话对话，时长共计 180 小时每组对话的标注信息包括转录文本、语音活动时间戳、说话人信息、录制信息和话题信息。说话人信息包括了性别、年龄和地域，录制信息包括了环境和设备。录制过程中环境相对安静。所有录制的参与者都是以中文为母语的流利普通话使用者，他们的口音略有不同。

AISHELL-1[18]：

400 名来自中国不同口音区域参与录制，经过专业语音校队人员转写标注并通过严格质量检验，数据库文本的正确率在 95%以上。录制时长 178 小时。

Aidatatang：

语料库长达 200 小时，邀请来自中国不同重点区域的 600 名演讲者参加录音，录音是在安静的室内环境或环境中进行，其中包含不影响语音识别的背景噪音。参与者的性别和年龄均匀分布。语料库的语言材料是设计为音素均衡的口语句子。每个句子的手动转录准确率大于 98％。

Primewords Chinese Corpus Set 1：

Primewords 包含了大约 100 小时的中文语音数据，这个免费的中文普通话语料库由上海普力信息技术有限公司发布。语料库由 296 名母语为英语的智能手机录制。转录准确度大于 98％，置信水平为 95％。抄本和话语之间的映射以 JSON 格式给出。

# 4 总结

相较于传统的文本纠错，语音文本有着其特有的特点。语音文本的错误更多的来源于不同人的发音问题。不同人的发音情况导致一般情况下语音文本更多的是同音字错误。学术界先将处理普通文本的方法迁移到了语音文本之中，但是语音文本的特殊性导致纠错率较低。语音文本更需要的是对同音字的纠错。需要对语音文本进行变长预测。

传统模型通过自回归模型利用序列之间的关系进行解码，得到了较高的纠正率。现在企业由于自回归模型线上部署的时延较高，从而转向非自回归模型，一次性解码，

加快解码速度。但是非自回归模型纠正率相较于自回归模型较低。随着学术界和企业界的不断研究，非自回归模型的纠正率在一步步地上升。在降低时延的同时尽可能的提高了纠正率。

总的来说不一定需要纯自回归或者非自回归，现今以及有实验将两者融合在一起使用，ASR 文本的纠错未来有着无限的可能。

# 参考文献

[1]Dagao Duan,Shaohu Liang,Zhongming Han and Weijie Yang.Pinyin as a feature of neural machine translation for Chinese speech recognition error correction.In First Online,2019.

[2]Marjan Ghazvininejad, Omer Levy, Yinhan Liu, Luke Zettlemoyer.Mask-Predict: Parallel Decoding of Conditional Masked Language Models.arXiv:1904.09324,2019.

[3]Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling Error Correction with Soft-Masked BERT. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 882–890, Online. Association for Computational Linguistics.

[4]Yisheng Xiao, Lijun Wu, Junliang Guo, Juntao Li, Min Zhang, Tao Qin, Tie-yan Liu.A Survey on Non-Autoregressive Generation for Neural Machine Translation and Beyond.arXiv:2204.09269,2022.

[5]Geoffrey Hinton,Oriol Vinyals,Jeff Dean.Distilling the Knowledge in a Neural Network.arXiv:1503.02531,2015.

[6]Graves A, Fernández S, Gomez F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]//Proceedings of the 23rd international conference on Machine learning. 2006: 369-376.

[7]Chris Hokamp and Qun Liu. 2017. Lexically Constrained Decoding for Sequence Generation Using Grid Beam Search. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

[8]Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-Based n-gram Models of Natural Language. Computational Linguistics, 18(4):467–480.

[9]Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling Error Correction with Soft-Masked BERT. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 882–890, Online. Association for Computational Linguistics.

[10]Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. 2021. Correcting Chinese Spelling Errors with Phonetic Pre-training. In Findings of the Association for Computational Linguistics: ACL-IJCNLP

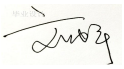
2021, pages 2250–2261, Online. Association for Computational Linguistics.
[11]Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova.BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.arXiv:1810.04805,2019
[12]Zhao Guo,Yuan Ni,Keqiang Wang,Wei Zhu and Guotong Xie.Global Attention Decoder for Chinese Spelling Error Correction.In ACL-IJCNLP,1419–1428,2021.
[13]Zheng Fang, Ruiqing Zhang, Zhongjun He, Hua Wu and Yanan Cao.Non-Autoregressive Chinese ASR Error Correction with Phonological Training.In NAACL,2022.
[14]Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K.0.Li,and Richard Socher. 2018. Non-autoregressive neural machine translation. In 6th International Conference on Learning Representations, ICLR 2018.OpenReview.net.
[15]Ruiqing Zhang, Xiyang Wang, Chuanqiang Zhang, Zhongjun He, Hua Wu, Zhi Li, Haifeng Wang, Ying Chen, and Qinfei Li. 2021b. BSTC: A large-scale Chinese-English speech translation dataset. In Proceedings of the Second Workshop on Automatic Simultaneous Translation.
[16]Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiang-Yang Li, Ed Lin, and Tie-Yan Liu. 2021. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition. In 35th Conference on Neural Information Processing Systems, NeurIPS 2021.
[17]Zehui Yang, Yifan Chen, Lei Luo, Runyan Yang, Lingxuan Ye, Gaofeng Cheng, Ji Xu, Yaohui Jin, Qingqing Zhang, Pengyuan Zhang, Lei Xie, Yonghong Yan.Open Source MagicData-RAMC: A Rich Annotated Mandarin Conversational(RAMC) Speech Dataset.arXiv:2203.16844,2022.
[18]Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, Hao Zheng.AISHELL-1: An Open-Source Mandarin Speech Corpus and A Speech Recognition Baseline.arXiv:1709.05522,2017.

# 中期检查表

## 苏州大学本科生毕业设计（论文）中期进展情况检查表

学院（部）：计算机科学与技术学院

<table>
<tr><td colspan="2">学生姓名</td><td>张翔</td><td>年级</td><td>2019</td><td>专业</td><td>计算机科学与技术</td><td>填表日期</td><td>2023/4/7</td></tr>
<tr><td colspan="2">设计（论文）选题</td><td colspan="7">面向语音识别文本的纠错系统实现</td></tr>
<tr><td rowspan="2">已完成的任务</td><td colspan="8">任务书、文献翻译、文献综诉以及完成。代码框架搭建完成。</td></tr>
<tr><td colspan="2">是否符合任务书要求进度</td><td colspan="6">是</td></tr>
<tr><td rowspan="2">尚须完成的任务</td><td colspan="8">将拼音编码加入到模型之中，调试模型。</td></tr>
<tr><td colspan="2">能否按期完成任务</td><td colspan="6">是</td></tr>
<tr><td rowspan="2">存在的问题和解决办法</td><td>存在的问题</td><td colspan="7">目前对自己的方法还有一些疑惑，有些实现细节还是不太懂。</td></tr>
<tr><td>拟采取的办法</td><td colspan="7">找相关资料理解方法。同时将模型细节方面弄懂</td></tr>
<tr><td colspan="2">指导教师意见</td><td colspan="7">抓紧进度，理清思路。<br>签名：</td></tr>
</table>

| | |
|---|---|
| 中期检查专家组意见 | 同意<br>组长签名： |
| 学院（部）意见 | 同意<br>教学院长签名： |

检查日期： 2023 年 4 月 10 日

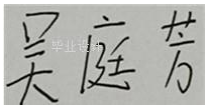
# 答辩记录表

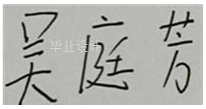
## 苏州大学本科生毕业设计（论文）答辩记录表

学院（部）：计算机科学与技术学院

| 学生姓名 | 张翔 | 年 级 | 2019 | 专 业 | 计算机科学与技术 |
|---|---|---|---|---|---|
| 设计（论文）题目 | 面向语音识别文本的纠错系统实现 | | | | |
| 答辩时间 | 2023 年 5 月 15 日 13：30 至 17：00 | | 答辩地点 | 理工楼 211 | |
| 答辩小组成员：<br>许佳捷，陈越，谷飞，吴庭芳 | | | | | |
| 答辩中提出的主要问题及学生回答问题的简要情况：<br><br>1. 模型中的拼音是怎么来的？<br><br>我们采用的 python 带的包 pypinyin，可以生成有声调的拼音。但是由于不同语境下，一个字的发音可能不同，我们也可以获得一个对应的拼音-文本文件，此文件中包含每个词语对应的拼音信息。我们可以对我们的文本采用最大词匹配的方式，将句子中的词一个个匹配，然后获得对应的拼音信息。<br><br>2. 怎么处理句子中的同音字情况。<br><br>拼音信息无法处理文本中的同音字问题，我们需要其他的策略来处理同音字问题，我们可以采用某些匹配的方式。我们可以采用候选词表对文本中的字进行相似度匹配，当某些词语和词表中的词语差不多且拼音相同时，我们可以将其中的词语替换为我们词表中的一些词，从而处理文本中的同音字问题。<br><br>3. 除了简单的给拼音进行编码，还可以采取什么方法加强拼音信息。<br><br>在本文中将拼音信息通过 torch.nn.Embedding 进行简单的编码，接着通过线性层赋予拼音信息一定的权重。但是有时候没有必要考虑到全部文本的拼音信息，我们只需要关注错误字符的拼音信息。所以我们可以采取 mask 掩码的方式对拼音编码进行处理。我们可以将正确字符的拼音编码 mask 掉，只考虑错误字符的拼音信息。这样可以加强我们对错误字符的识别。将 mask 拼音与检测标签的线性层进行联合训练，只对检测标签中当前标签为错误的字符加上拼音进行加强训练。 | | | | | |

答辩小组组长签名：

答辩小组成员签名：，，

记录人签名：

2023 年 5 月 15 日

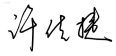
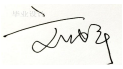
# 成绩评定表

## 苏州大学本科生毕业设计（论文）成绩评定表

学院（部）：计算机科学与技术学院

<table>
<tr><td colspan="4">设计（论文）题目：面向语音识别文本的纠错系统实现</td></tr>
<tr><td>姓 名</td><td>张翔</td><td>学 号</td><td>1927405126</td></tr>
<tr><td>年 级</td><td>2019</td><td>专 业</td><td>计算机科学与技术</td></tr>
<tr><td>指导教师评语</td><td colspan="3">张翔同学的毕业内容是实现一个面向语音识别文本的文本纠错系统，主要思路是基于目前已有的 seq2edit 纠错模型，融入拼音信息，以提高语音识别产生的文本上的纠错性能。张翔同学学习态度好、做事情有责任心，努力完成每一项任务，比较完满地完成了预定目标。论文书写规范、认真。</td></tr>
<tr><td></td><td colspan="2">评 价 内 容</td><td>得 分</td></tr>
<tr><td></td><td colspan="2">设计（论文）方案设计、文献检索、阅读及综述能力、进度等情况评价分（计 25 分）</td><td>21</td></tr>
<tr><td></td><td colspan="2">毕业设计（论文）质量和工作量评价分（计 50 分）</td><td>43</td></tr>
<tr><td></td><td colspan="2">科学素养、学习态度、纪律表现等情况评价分（计 25 分）</td><td>23</td></tr>
</table>

<table>
<tr><td></td><td>成绩（满分 100 分）：87　　签名：　　2023 年　　5 月　　15 日</td></tr>
<tr><td>评阅教师评语</td><td>本文对语音识别文本纠错领域中一些经典模型进行了系统的分析和研究，实验证明，融合了拼音信息的 GECToR 模型相较于传统的 GECToR 模型性能更优，同时标签词表对模型的纠错能力也有重要的影响。该生对于研究课题能熟练运用本专业所必需的基础理论和专业知识，分析问题，解决问题；理论正确，概念比较清楚，应用合理，论文撰写较为清晰，通顺，达到了毕业设计任务书的相关要求。<br><br>
<table>
<tr><td>评　价　内　容</td><td>得 分</td></tr>
<tr><td>毕业设计（论文）文字书写评价分（计 20 分）</td><td>18</td></tr>
<tr><td>毕业设计（论文）质量评价分（计 40 分）</td><td>37</td></tr>
<tr><td>工作量情况评价分（计 20 分）</td><td>18</td></tr>
<tr><td>毕业设计（论文）创新及分析问题、解决问题能力评价分（计 20 分）</td><td>17</td></tr>
</table>
成绩（满分 100 分）：90　　签名：　　2023 年　　5 月　　15 日</td></tr>
<tr><td>答辩小组评语</td><td>该生对本文对语音识别文本纠错领域中的模型进行分析研究，融合了拼音信息用于纠错，实验证明，相较于传统的 GECToR 模型性能更优。论文能够简明扼要、有重点突出地阐述主要内容，论点基本正确，思路较清晰，但是存在一定格式问题。能够较好的回答老师的问题，有一定的理论根据，思路一般，论点较为正确，概念基本清楚，按照毕业设计任务书，完成了任务书所规定的各项要求。<br><br>
<table>
<tr><td>评　价　内　容</td><td>得 分</td></tr>
<tr><td>毕业设计（论文）介绍表达情况评价分（计 20 分）</td><td>17</td></tr>
<tr><td>回答问题表现评价分（计 40 分）</td><td>31</td></tr>
<tr><td>毕业设计（论文）水平和工作量评价分（计 40 分）</td><td>30</td></tr>
</table></td></tr>
</table>

| | |
|---|---|
| | 成绩（总分 100 分）：78　组长签名：　2023 年　5　月 15 日 |
| 按权重折算总成绩 | 84 |
| 审定成绩：84 | 答辩委员会主任签名：<br>日　　期： |

# 检测报告

## 文本复制检测报告单

№:ADBD2023R_20181127141145202305261939038143641685 23

检测文献： 面向语音识别文本的纠错系统实现

作者： 张翔

检测范围：
中国学术期刊网络出版总库
中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库
中国重要会议论文全文数据库
中国重要报纸全文数据库
中国专利全文数据库
图书资源
优先出版文献库
大学生论文联合比对库
互联网资源(包含贴吧等论坛资源)
英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)
港澳台学术文献库
互联网文档资源
CNKI大成编客-原创作品库

时间范围： 1900-01-01至2023-05-26

检测时间： 2023-05-26 19:39:03

总文字复制比：2.8%

去除引用：2.8% 去除本人：2.8% 重合字数：519 文献总字数：18609

单篇最大文字复制比：1.3% 单篇最大重复字数：249

总段落数： [ 6 ] 疑似段落数： [ 2 ] 疑似段落最大重合字数： [ 440 ]

前部重合字数： [ 79 ] 后部重合字数： [ 440 ] 疑似段落最小重合字数： [ 79 ]

| | |
|---|---|
| 2.2% | 中英文摘要等（总3650字） |
| 14.8% | 第一章绪论（总2968字） |
| 0% | 第二章基础知识概述（总3992字） |
| 0% | 第三章融合拼音信息的GBCTcR模型（总3583字） |
| 0% | 第四章实验过程和结果分析（总3208字） |
| 0% | 第五章总结与展望（总1208字） |

### 中英文摘要等

总文字复制比：2.2%（79） 总字数：3650

| | | |
|---|---|---|
| 1 | 建筑给排水节水节能技术和实施策略<br>薛卫东; - 《中国高新科技》- 2021-01-25 | 1.0%<br>是否引用：否 |
| 2 | 基于大数据分析的聊天机器人研究<br>张帅;刘海林;王媛媛; - 《数字通信世界》- 2020-01-01 | 1.0%<br>是否引用：否 |
| 3 | 虚拟现实技术在毒品预防中的应用现状及未来构想 | 0.9% |

中国知网文献检测报告单（机构版）