北京大学

硕士研究生学位论文

题目：**基于深度学习的语音识别技术在智慧教室中的应用与实现**

姓　名：	苏凤宇
学　号：	2001210415
院　系：	软件与微电子学院
专　业：	软件工程
研究方向：	自然语言处理
导师姓名：	蒋严冰 副教授

二〇二三 年 三 月

版权声明

摘要

基于现代教育信息化的发展需求，以及当前疫情形势的现实因素，智慧教室，在线学堂等线上教育应用越来越成为教育教学场景中的重要组成部分。但是由于当前网络环境不稳定以及学生听课场景复杂等因素影响，会极大降低学生听课效率和使用体验，而在学生终端加入字幕可以很好的解决这类问题，帮助学生更好的理解教学内容。随着当前深度学习技术的深入发展与落地，让基于深度学习的语音识别正确率在特定场景下可以达到人类水平，这就为自动字幕生成奠定了基础。基于上述现实需求和意义，本文将使用基于深度学习的语音识别技术实现在智慧教室场景下的自动字幕生成，同时提高教师教学和学生学习的效率。

本文主要使用基于深度学习的端到端语音识别技术实现智慧教室场景下的自动字幕生成功能。首先使用开源的 Wenet 语音识别框架和教育场景的语音识别数据集训练基础的语音识别模型，然后基于 Libtorch 对此模型推理代码进行 c++实现，同时提供流式识别和非流式识别两种识别模式，丰富了服务请求的灵活性，且极大提高模型推理的时间性能，从而能够实现教师直播课程的实时字幕生成功能，并且后端加入通用语言模型和热词模型进行辅助解码，极大提高在特定场景下的语音识别准确率。在模型部署上，本文使用 Docker 容器技术对训练好的语音识别模型进行服务封装部署，并实现了支持多种语言调用的 Grpc 远程过程调用接口，使语音识别服务端和请求端在语言层面和设计层面解耦，方便模型服务的移植部署。最后在字幕生成阶段，字幕生成模块将智慧教室客户端发来的流式语音信号通过流式模型服务请求获得原始的语音识别结果文本，之后对此原始文本进行预处理，主要包括使用预先训练好的模型进行文本纠错和添加标点操作，进一步提高自动生成字幕的准确率和可读性，然后通过时间戳对齐和字幕缓存技术实时生成字幕帧数据，并将其发送给智慧教室客户端进行最终的渲染和显示。

在上一代智慧教室自动字幕生成模块的基础上，本文主要通过模型算法改进和工程性能优化实现了在线直播课程的实时字幕生成，丰富了自动字幕技术的应用场景，提高了智慧教室项目的功能完备性，也为后续基于语音识别服务的高阶功能扩展开发提供了基础。

关键词：深度学习，语音识别，语言模型，自动字幕，智慧教室

Application and Implementation of Deep Learning-based Speech Recognition Technology in Smart Classrooms

Fengyu Su (Software Engineering)
Directed by Yanbing Jiang

ABSTRACT

Based on the development needs of modern education and the real factors of the current epidemic situation, online education applications such as smart classrooms and online learning classes are becoming an important part of the education teaching scene. However, due to the current unstable network environment and the complexity of students' listening scenarios, it can greatly reduce students' listening efficiency and experience, while adding subtitles to students' terminals can well solve such problems and help students better understand the teaching content. With the deep development and implementation of current deep learning technology, the correct rate of speech recognition based on deep learning can reach human level in specific scenarios, which lays the foundation for automatic subtitle generation. Based on the above realistic needs and significance, this paper will use deep learning-based speech recognition technology to realize automatic subtitle generation in smart classroom scenarios and improve the efficiency of teachers' teaching and students' learning at the same time.

In this paper, we mainly use deep learning-based end-to-end speech recognition technology to implement automatic subtitle generation in smart classroom scenarios. Firstly, we use the open source Wenet speech recognition framework and speech recognition dataset for educational scenarios to train the basic speech recognition model, and then implement this model inference code in C++ based on Libtorch, which enriches the flexibility of service requests and greatly improves the time performance of model inference, thus enabling real-time subtitle generation for teachers' live courses, and the back-end incorporates a common language model and a hot word model to assist in decoding, which greatly improves the speech recognition accuracy in specific scenarios. In terms of model deployment, this paper uses Docker container technology to deploy the trained speech recognition model in

a service package, and implements a Grpc remote procedure call interface that supports multiple language calls to decouple the speech recognition server and request side at the language level and design level, which facilitates the porting and deployment of the model service. Finally, in the subtitle generation stage, the subtitle generation module takes the streaming speech signal from the smart classroom client and requests the original speech recognition result text through the streaming model service, and then pre-processes this original text, mainly including error correction and punctuation operations using the pre-trained model to further improve the accuracy and readability of the automatically generated subtitles, and then generates the subtitle frame data in real time through timestamp alignment and Then the caption frames are generated in real time through timestamp alignment and caption caching technology and sent to the smart classroom client for final rendering and display.

Compared to the previous generation of automatic subtitle generation module for smart classroom, this paper mainly implements real-time subtitle generation for online live courses through model algorithm improvement and engineering performance optimization, which enriches the application scenarios of automatic subtitle technology, improves the functional completeness of the smart classroom project, and provides a basis for the subsequent development of advanced functions based on speech recognition services.

KEY WORDS: Deeplearning, Speech recognition, Languare model, Automatic subtitle generation, Smart Classroom

目录

第一章 绪论

1.1 研究背景

1.1.1 在线教育的发展趋势

随着社会信息化技术的大力发展和普及，以及人们对子女教育的重视程度越来越

高，并且叠加近几年疫情防控政策等现实因素的影响，教育形式多样性和高效性的需求越来越凸显。

而在线教育作为一种新型的教育模式，在近几年来发展迅速，且相对于传统线下教育优势明显，首先就是时间和地点的灵活性，学生可以自由安排学习时间和地点，其次可以适应不同学习风格的需求，在线教育提供了多种学习方式，如视频课程、自学课程、讨论组、实验室活动等，可以满足不同学习风格的学生的需求，最重要的一点就是可以提高教师的教学质量，线教育可以使用最新的教学资源和技术，例如，可以使用虚拟实验室、模拟软件、在线测验等工具来提高学生的学习效果。

基于在线教育的种种优点，中国的在线教育渗透率也在不断扩大。根据 2021 年 1 月 20 日教育部等五部门联合发布的《关于大力加强中小学线上教育教学资源建设与应用的意见》，可以看出国家也在大力引导教育信息化的发展趋势，也为在线教育的规范健康发展提供了保障。

而本文的智慧教室项目，即是一种基于各种人工智能技术赋能的在线教育平台，旨在用技术的手段提高教师远程授课的教学质量，比如应用计算机视觉技术可以实时监测学生上课状态并进行适当提醒，基于自然语言处理的技术可以提取教师课程内容的关键词辅助教学，也可以自动检查随堂测验的结果并进行统计标注等，这些人工智能技术的开发应用可以极大的方便教师的上课效率，提高学生的听课质量。

1.1.2 远程直播教学存在的问题

虽然智慧教室等在线教育平台有很多优点，但同时也存在很多现实的问题，比如缺乏直接的师生互动，容易被听课的电子设备上的其他信息推送干扰，以及容易受到

当前的网络环境以及现实环境噪声影响等，而这些问题有的可以通过与家长学生沟通适当约束处理，有些则可以通过技术手段解决。本文就是旨在通过使用语音识别的技术实时的生成教师授课内容的字幕，可以很大程度的缓解因学生听课环境嘈杂或者跟不上授课节奏而降低听课效率的影响。而其他的一些问题还需要在线教育的从业者们继续去努力克服解决。

1.2 国内外研究现状

1.2.1 字幕技术研究现状

字幕技术是一种将文本内容显示在视频上的技术，通常用于将另一种语言的音频翻译成文本，或者为视频提供补充信息。字幕技术的研究可以追溯到 20 世纪 60 年代，随着计算机和互联网的发展，字幕技术也得到了极大的提升。近年来，字幕技术发展迅速，许多研究机构和企业都在投入巨资进行字幕技术的研究和开发。

当前字幕生成方案大致分为两大类，一类是采用人工识别并标注最后校验的方案，一般是在准确率要求较高的离线场景下采用，比如一些电视剧电影的字幕制作就是采用此种方案，但是此方案效率较低，需要耗费很高的人力成本和时间成本。另一类就是目前很受欢迎的自动字幕生成方案，随着当前语音识别、机器翻译和文本生成等方面的技术的进步与成熟，使自动的识别声音生成对应的文本字幕成文可能，此方案效率较高，但也分为离线和在线两种模式，且支持在线识别场景，能够满足一些比如视频会议，直播课程等实时性要求较高的场景，所以也越来越成为目前主流的研究方向。

关于在线教育领域，比如国内的网易公开课，慕课等课程平台目前也都开始支持录播课程讲述内容的自动字幕生成，极大的减少了教师的课程制作成本，可以把更多

的精力用在课程内容本身，提高教学质量。而对于直播课程，受限于计算资源，算法性能等因素的影响，目前还没有成熟的解决方案，还需要研究者们探索研究，这也是本文的主要研究内容。

1.2.2 语音识别技术研究现状

语音识别是一种将语音转换为文本的技术，最早可以追溯到上世纪 50 年代。当时，贝尔实验室的研究人员就开始尝试使用电子计算机来处理语音信息，主要是通过程序规则识别简单的词语并将其组合为文本，这也为接下来的 20 年基于规则的孤立词识别系统的研究奠定了开端。

到了 20 世纪 80 年代，语音识别系统的研究开始从孤立词的识别转向大量连续语音识别的研究。统计语言模型大行其道，将时序建模和观察概率建模相结合的 GMM-HMM 成为研究的主流，但是由于当时的技术研究还处于初期，可以用来训练的标注数据也不是很充足，所以识别的准确率还很低，无法达到商用的水平。

20 世纪 90 年代到 2010 年间，随着对 GMM-HMM 的研究发展逐渐走向成熟，商用的语音识别系统开始出现，例如 DRAGON 系统，IBM 的 Via-vioce 系统，微软的 Whisper 系统，英国剑桥大学的 HTK 系统等，尤其是开源的 HTK 系统，提供了一整套语音识别的软件工具，为语音识别的生态发展起到了很大的贡献。

进入 2010 年之后，由于深度学习，循环神经网络的重新兴起，基于深度学习的语音识别系统的技术研究实现了飞跃发展。从基于 DNN-HMM 开源语音识别工具 Kaldi 到近些年出现的端到端的解决方案，比如 DeepSpeech2，Espnet，Wenet 等，语音识别模型的识别准确率不断提高，各个模型在某些特定数据集上都取得了很好的甚

至超过了人类水平的测试结果，但是这些数据集是由专业人士在干净的环境下录制的，和实际的工业使用场景还是有很大差异，要想将其应用部署在实际复杂多变的智慧教室场景下，并取得可以实用的效果，还是要面临很大挑战。

1.3 研究的主要内容及意义

本文主要以智慧教室项目为依托，通过语音识别技术自动的为教师直播课程生成文本字幕，提高授课效率，并通过开发相应的服务接口，灵活的集成到智慧教室产品中，提高其功能完备性，也为后续基于语音识别技术的其他功能扩展提供服务基础。

本文的研究过程主要包括以下几个方面。

1.3.1 语音分割

通常对于输入的语音信号，会有很多无用的空白信号，有意义的人声信号片段只占一部分，语音分割是指使用信号特征提取和端点检测等技术将语音信号分割成若干个连续的话语片段，并且能够去除空白信号，减轻后端识别的压力。本文主要是基于 webrtcvad 模块进行静音帧检测，并根据检测的结果设计相应的算法逻辑对有效帧进行分割提取。

1.3.2 语音识别

本文的语音识别模块主要是基于开源的 Wenet 语音识别系统框架，在其提供的基于开源的 Wenetspeech 数据集预训练模型的基础上，在教育领域语音识别数据集上进行微调训练，同时加入通用语言模型和热词模型对候选结果进行加权打分，最终筛选出最优结果。并且实现了字级别时间戳对齐和语音信号端点检测，从而使其支持连续

长语音信号的实时识别，为后续的实时字幕生成提供技术基础。

1.3.3 模型部署

为了提高模型的高可用性，提供稳定的语音识别服务，并应用在智慧教室产品中，本文使用效率更高的 c++语言重新实现了模型推理和解码算法，减少语音识别的时间开销，同时基于 Grpc 将语音识别服务包装成远程调用接口，增强用户调用服务的灵活性，最终使用 Docker 容器技术将代码模型和环境打包成镜像文件，根据需求在合适机器上移植部署。

1.3.4 文本处理和字幕生成

对于语音识别模块识别的原始文本，通常包含很多错别字词以及一些不连贯的情况，本文使用无标注的通用语料训练了一个中文纠错模型对其进行纠错处理，进一步提高字幕生成的准确率。另外识别的结果文本是不包含标点符号的，所以本文通过使用语音分割技术和标点添加模型相结合，为得到的文本分句并添加合适的标点符号，提高最终生成字幕的可读性和连贯性。最终将处理好的文本字幕实时发送给客户端渲染显示。

1.4 拟解决的关键问题

1.模型推理时间开销大，不支持在线识别的问题：基于 LibTorch 库使用 C++语言重写模型推理代码，并将 pytorch 框架训练生成的模型导出成 LibTorch 可加载的格式，以支持模型加载和在线推理。

2.不支持实时长语音识别问题：通过语音识别模块内部集成语音端点检测模块，

能够识别用户完整语句的停止位置，并据此为依据输出完整连续的用户语音识别结果。

3.输出的语音识别结果存在错误字词和不连贯的问题：因为本文的字幕生成的应用场景是在单一的在线教育领域，会有很多领域常用词存在，所以可以使用热词增强技术，构建一个热词语境图，在解码搜索阶段，加入热词前缀序列得分增强。同时支持语言模型，使用先验的语言知识对结果进行打分纠错。使用这两种方法可以很好的减少常用词识别错误率。

1.5 论文章节安排

第一章绪论，主要介绍了本文研究的相关时代背景和技术背景，字幕技术和语音识别技术当前的发展现状，并结合智慧教室中自动字幕的现实需求，简单介绍了本文研究的各个方面和主要解决的问题，梳理出本文的主要研究过程。

第二章系统设计概要和相关技术简介，先是系统分析了本文研究内容的各项需求，之后展示了本文的系统结构设计和数据流设计，最终系统的介绍了各个功能模块用到的相关技术和实现逻辑。

第三章语音识别，主要介绍了本文用到的语音识别框架及其实现算法，开源数据集整理及其训练过程，另外重点讲述了为了提高模型识别准确率而加入的语言模型和一些相关辅助功能实现，是本文的重点算法章节。

第四章模型服务与部署，主要介绍两种模型推理方案的对比和选择，训练好的模型到处和量化方式，以及语音识别模块对外的服务接口设计，最终介绍了 Docker 容器环境部署流程。

第五章自动字幕生成，是本文的功能统领章节，按照数据流的顺序，逐节阐述了

从原始的语音信号输入到最终生成字幕的各个算法实现环节。

第六章系统测试与分析，主要是配置本地环境，编写测试样例，对本文实现的自动字幕模块进行功能和性能测试，并对测试结果进行系统分析和总结。

第七章研究总结与启示，主要分析了本文研究过程中存在的一些问题和一些解决办法，以及基于此研究未来的一些改进完善的方向。

第二章 系统设计概要

2.1 系统需求分析

智慧教室是我们研发的一个在线教育平台，其主要的特点就是大量使用人工智能的技术开发出各种自动化辅助教学工具，提高线上授课效率，增强教师和师学生的沉浸式授课体验，既能实现线上教室的功能性，又能体现快捷高效的智慧性，这也是我们智慧教室项目研发的初衷。本文的自动字幕技术就是在这种初衷的指引下，解决课程中缺乏字幕影响授课效率的痛点，提高智慧教室的功能完备性。

而作为一种线上教育基于平台，基本功能就是提供教师授课和学生听讲的一种连接平台。而这种连接包括离线课程录播和在线远程直播两种方式，相应的本文的自动字幕模块也分为离线课程自动字幕生成和直播课程实时字幕显示两大基本需求。而具体的细分需求会在下述章节中详细阐述。

2.1.1 功能性需求分析

本文的自动字幕模块的总体功能需求字幕请求和字幕返回功能，音频信号处理功能，语音识别功能，文本加工功能。

图 2.1 为自动字幕模块整体需求用例图。

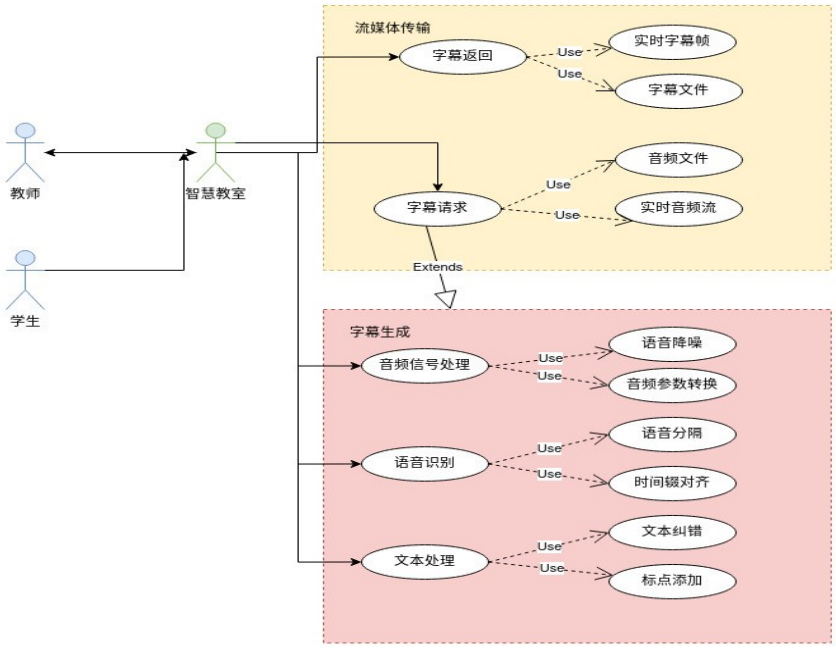


图 2.1 自动字幕模块整体需求用例图

1.流媒体传输功能需求分析

此功能属于字幕生成模块和智慧教室系统的通信接口，主要需求是实现字幕生成服务的请求调用和结果返回，并保证数据传输的稳定性和可靠性。

根据智慧教室业务需求逻辑不同，主要分为音频文件请求和实时音频数据流请求。

系统会定时监测智慧教室数据存储服务器中的课程变化情况，若检测到有新增的教师课程视频时，会自动的发起请求调用，对新增课程进行音视频分离，并将提取的

音频文件发送给自动字幕模块，同时，教师客户端侧也可以主动发起字幕生成请求，自动字幕模块通过内部的软件算法生成对应的字幕文件，之后将结果发送回智慧教师后端以及教师客户端。

若智慧教室启动一次直播课程，系统后端会将直播音频信号实时转发给自动字幕模块，自动字幕模块生成字幕文本并进行时间戳对齐后按照帧格式发送给系统以及客户端进行实时的渲染显示。

2.字幕生成功能需求分析

（1）音频信号处理需求

由于音频录制设备的差异或者设置模式的原因，音频信号的通道数和采样率一般不同，而后端语音识别模块要求的输入信号一般为固定参数，所以需要根据具体设定进行信号的参数对齐转换。另外，原始的语音信号中通常会包含很多教室上课的环境噪声以及一些其他人的干扰信号，同时在远程传输中也会加入一些噪声信号，为了提高模型识别的准确率，我们会使用语音信号处理算法进行降噪和人声提取处理，提高模型输入信号的信噪比。

（2）语音识别需求

此需求是本模块的核心功能需求，为了提高语音识别算法的泛化能力，要使用大量语音识别数据集训练高效的算法模型，并且加入辅助的逻辑规则，生成稳定的语音识别文本结果。由于本文支持离线和在线字幕生成两种生成模式，所以要求语音识别算法也能支持流式和非流式两种信号输入模式。同时为了支持连续的长语音输入，所以算法内部还需有语音分割和端点检测的功能，并且需要输出识别结果文本对应的时间戳，以便满足后续文本处理的需要。

（3）文本处理需求

在深度学习和大数据的加持下，语音识别算法识别的文本结果虽然正确率得到了很大提高，但还是会有很多错误字词和语义不连贯的情况，所有需要使用一些通用的语言纠错模型进行错误字词检查和纠正。另外，得到的文本是不包含标点符号的，这对于书面文本的阅读很不友好，所以还需要通过端点检测和对文本语境的理解在合适的位置添加合适的标点符号，提高生成字幕的可读性。

2.1.2 非功能性需求分析

本系统研发后会最终应用在智慧教室产品中，所以除了实现基本功能之外，还需要考虑产品功能稳定性，部署成本和维护成本以及对未来新业务需求的扩展支持。

1.稳定性和高可用性

稳定性和鲁棒性作为业务软件开发的基本要求至关重要，首先需要支持各种异常处理，包括信号输入格式不一致，通信中断恢复等情况的处理。其次是进行冗余设计，若发现处理结果异常时，尝试更换数据处理链路。最重要的要有服务自检机制，如发现自身服务异常或停止时，尝试自动重启恢复，若无法恢复需向智慧教室后端发送服务停止消息和日志，方便排查维护。

2.响应实时性

对于离线字幕生成请求，一般对实时性要求不高，对于多个请求同时发送的情况，可以使用请求队列或者多进程多服务的方式满足需求。而对于实时字幕生成请求，则需要考虑算法处理延迟问题，若延迟较大会极大影响用户体验甚至失去字幕生成功能

存在的意义，所以需要使用模型量化和工程优化等技术较少延迟，提高用户需求相应实时性。

3.可扩展性

在模块设计之初，就要考虑各个功能模块的解耦设计，在满足性能的前提下，各个功能尽量包装成服务，并在考虑到各种调用场景的情况下灵活的设计好服务接口，方便主业务逻辑层调用。同时为日后其他基于语音识别技术的业务需求开发提供基础服务，较少日后的重复开发工作量。

4.成本

自动字幕模块的附加成本包括部署成本和维护成本。通过程序算法优化，尽量做到模型部署轻量化，降低 cpu 和内存的资源消耗，并在部署时考虑使用负载均衡策略，优化运行成本。同时也要考虑模块的可维护性，开发中注意加入注释和说明文档整理，系统中加入分级日志机制，总结出异常恢复的常用办法，方便功能模块的部署维护和异常处理。

2.2 系统架构设计

基于上述章节描述的各种功能和性能需求，本文基于软件工程系统架构的设计方法，对字幕生成模块进行系统设计，使其方便的集成到智慧教室系统中。

智慧教室作为一种在线课程的网络平台，使用较为常见的客户端服务器的架构模式，客户端包括学生客户端和教师客户端，通过使用 WebRTC 实时音视频传输协议，将客户端采集的音视频信号发送给服务后端，后端的流媒体服务器集群会对原始的音

视频信号进行编码解码，之后调用各种 AI 服务进行后处理操作，最终将处理好的视频流推送给相应客户端，同时转存到文件存储服务器。

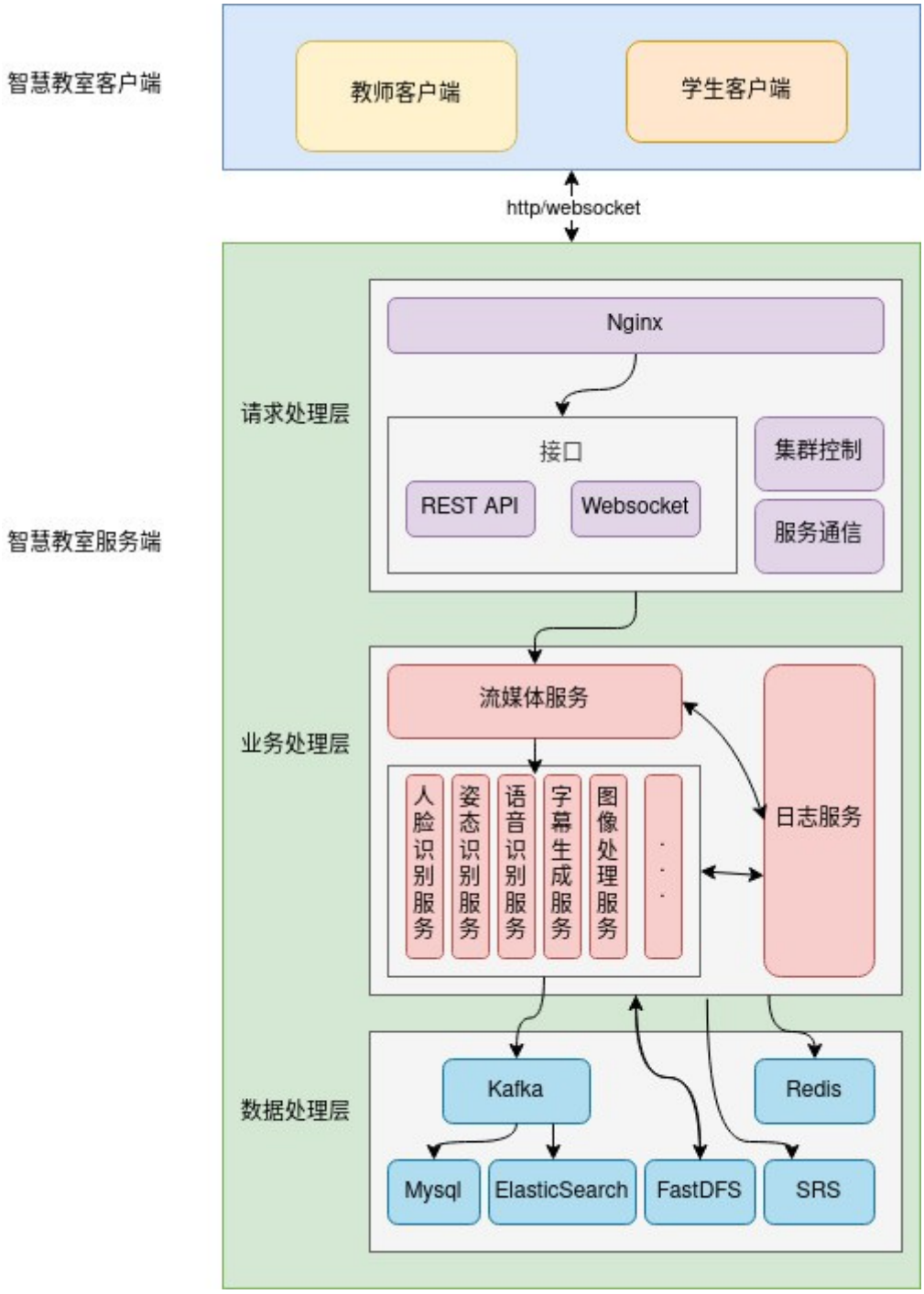
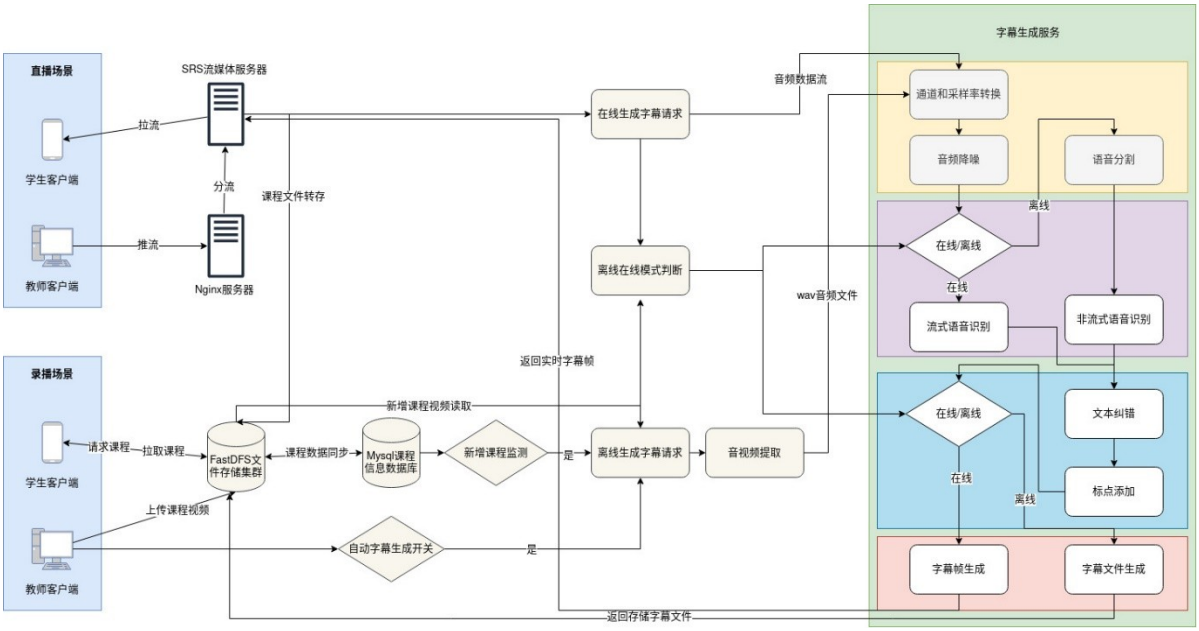


图 2.2 智慧教室系统架构图

本文主要工作是为智慧教室系统提供语音识别和字幕生成服务功能。其中客户端侧

使用开源的 Ckplayer 实现字幕的加载和播放工作，在智慧教室服务端，请求处理层主要是提供网络负载均衡的功能，使所有的流量得到合理分配，保证服务请求的及时相应。业务处理层为系统提供各种基础服务，包括最基本的完成音视频信号推流拉流工作的流媒体服务，智慧教室项目使用的流媒体传输技术包括 WebRTC 和 RTMP 两种传输方案，而流媒体服务层则会调用包括姿态识别，字幕添加等 AI 服务等进行处理，本文的主要内容就是设计在此层。最下层是数据处理层，首先设计了两个中间件模块，通过使用 Kafka 消息队列和 Redis 业务数据缓存的方式缓解系统数据读写压力，底层数据库则完成系统业务数据的存储功能，其中 Mysql 负责存储业务信息，用户信息，课程信息等元数据，而 FastDFS 文件存储集群则用来分布式存储体积较大的课程音视频数据。



2.3 数据流设计

根据智慧教室的业务需求，分为录播回放和课程直播两种业务场景，相应的自动

字幕生成服务也分为离线音频文件字幕生成和在线音频流字幕生成两条数据流模式。

图 2.3 为智慧教室自动字幕生成数据流图。

图 2.2 智慧教室自动字幕生成数据流图

对于离线录播课程的场景，教师把事先录制好的课程视频上传，智慧教室后端在重新解码编码等基本处理后将课程信息和数据存储在数据库和文件存储集群中，课程监测模块监测到新增课程标志后，在确认教师开启自动字模生成选项后，会向字幕生成模块发送离线字幕生成请求，并同时进行音视频分离操作，提取出 wav 格式的音频文件，然后调取字幕生成服务生成字幕文件，最终将字幕文件存储到 FastDFS 文件服务器中并更新数据库中的课程信息。之后学生若观看课程视频发起课程请求时，系统会将课程视频和字幕文件同时发送给客户端，由客户端播放器进行字幕的添加和渲染。

而对于教师直播授课的场景，教师的直播音视频流会推送到后端，首先经过 Nginx 负载均衡服务器进行分流操作，之后经过 SRS 流媒体服务器进行流媒体的处理，同时向字幕生成模块发起在线字幕生成请求，并将音频数据流实时转发给字幕生成服务，字幕生成服务会实时生成带有时间戳的文本字幕帧，最终再由流媒体服务器将字幕帧数据发给学生客户端进行实时显示。

2.4 本章小结

本章主要对自动字幕模块进行了全面的需求分析和系统架构设计。

第一节使用软件工程的方法对自动字模的功能需求和非功能需求进行了系统分析，从而为之后的系统实现提供清晰的指引。

第二节主要介绍了智慧教室项目的整体分层架构，以及本文的自动字幕模块在其

中处于什么位置。

第三节阐述了系统整体数据流的转移处理过程，详细展示了各个子功能模块的逻辑连接关系，向读者讲述了系统的整体工作流程。

第三章 语音识别

3.1 语音识别技术简介

语音识别作为本系统的核心模型算法，是实现自动字幕生成功能的基础。主要就是对输入的语音信号进行特征提取得到一个特征向量序列，再通过一些识别算法处理，得到对应的语音识别文本序列。

语音识别使用数学模型定义为：

$$W^{¿} \quad ¿\, argmax_w P(W \mid X) \quad (1)$$

$$¿ \quad \approx argmax_w \frac{P(X \mid W) P(W)}{P} \quad (3)$$

其中 W 表示结果文本序列，X 表示输入语音信号。公式 1 表示语音识别的优化目标是在给定语音输入的条件下，找到一个文本序列，使得条件概率结果最大。根据贝叶斯公式，可以得到公式 2，其中分母表示得到这条语音信号的自然概率，它的大小和优化目标没有参数关系，可以在求解时不予考虑，从而得到公式 3。公式 3 的第一部分表示给定一个文本序列，出现当前语音信号的概率，它就是语音识别的声学模型，也是语音识别算法的基本实现模型；第二部分则表示此文本序列出现的自然概率，就是语音识别中的语言模型，加入语言模型这个先验信息可以很大程度提高语音识别的准确率。

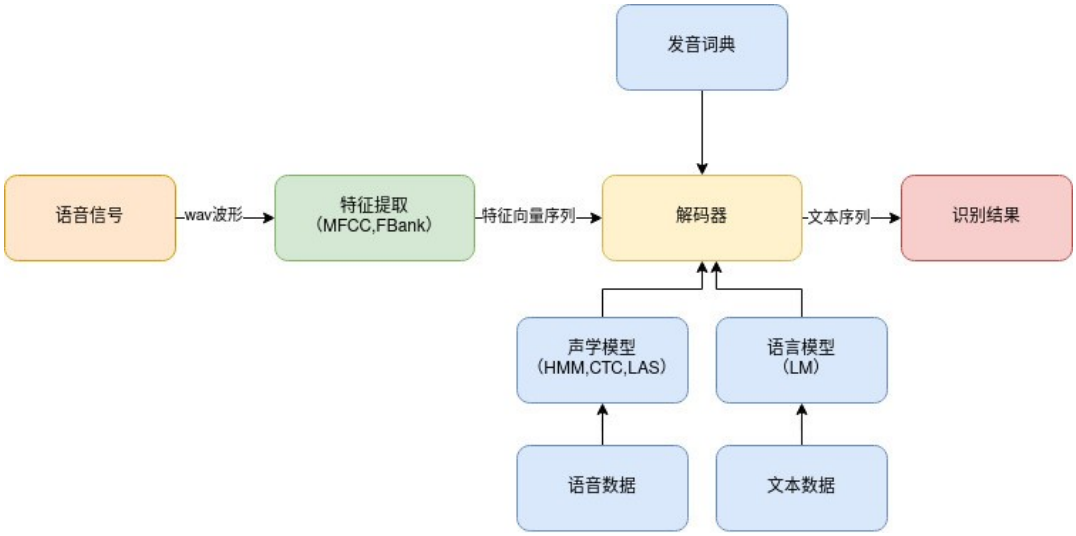


图 3.1 为语音识别系统基本流程图。

图 3.1 语音识别系统基本流程图

3.1.1 特征提取

本文主要讲述如何根据音频信号提取 MFCC 和 FBank 特征，这也是目前在语音识别任务中使用最广泛的两种特征。

人类的语音信号的频率大部分在 10kHz 以下，根据奈奎斯特采样定理，20kHz 的采样率就足够了。我们通常使用麦克风进行音频录制的采样率为 16kHz，一个采样点使用 16bit 来存储。

图 3.2 为特征提取流程图。

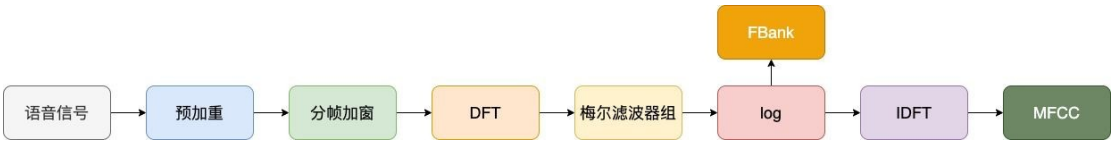


图 3.2 特征提取流程图

输入的语音信号，首先经过预加重处理，在音频录制过程中，高频信号更容易衰减，而像元音等一些因素的发音包含了较多的高频信号的成分，高频信号的丢失，可能会导致音素的共振峰并不明显，使得声学模型对这些音素的建模能力不强。预加重是个一阶高通滤波器，可以提高信号高频部分的能量。

第二步是分帧加窗操作，语音信号是一个非稳态的、时变的信号。但在 短时间范围内可以认为语音信号是稳态的、时不变的。这个短时间一般取 10-30ms，因此在进行语音信号处理时，为减少语音信号整体的非稳态、时变的影响，从而对语音信号进行分段处理，其中每一段称为一帧，帧长一般取 25ms。为了使帧与帧之间平滑过渡，保持其连续性，分帧一般采用交叠分段的方法，保证相邻两帧相互重叠一部分。相邻两帧的起始位置的时间差称为帧移，我们一般在使用中帧移取值为 10ms。因为后面会对信号做 FFT，而 FFT 变换的要求为：信号要么从-∞到+∞，要么为周期信号。现实世界中，不可能采集时间从-∞到+∞的信号，只能是有限时间长度的信号。由于分帧后的信号是非周期的，进行 FFT 变换之后会有频率泄露的问题发生，为了将这个泄漏误差减少到最小程度，我们需要使用加权函数，也叫窗函数。加窗主要是为了使时域信号更好地满足 FFT 处理的周期性要求，减少泄漏。

接下来的操作是 离散傅里叶变换（缩写为 DFT），将每个窗口内的数据从时域信号转为频域信号，从而更方便提取信号的标志性特征。DFT 的时间复杂度为$O(n|2)$,

为了满足系统的实时性要求，在实际中使用的是快速傅里叶变换（缩写为 FFT），其时间复杂度是$O(nlog(n))$。

之后是重要的梅尔滤波器处理，从 FFT 出来的结果是每个频带上面的幅值，然而人类对不同频率语音有不同的感知能力，频率越高，感知能力就越差。梅尔刻度（Mel）是一种非线性刻度单位，表示人耳对音高变化的感官，基于频率定义。在 Mel 频域内，人的感知能力为线性关系,能更好的符合物理世界基本原理。将梅尔域上每个三角滤波器的起始、中间和截止频率转换线性频率域，并对 DFT 之后的谱特征进行滤波，得到 40 个滤波器组能量，为了使特征对输入信号的扰动不敏感，再进行 log 操作，就得到了 40 维的 Fbank（Filter Bank）特征。

最后使用离散余弦变换（IDFT）对滤波器组系数去相关处理，并产生滤波器组的压缩表示。通常，保留所得到的 2-13 阶倒频谱系数，就是 MFCC 特征结果，其余部分被丢弃，丢弃其他系数的原因是它们代表了滤波器组系数的快速变化，并且这些精细的细节对语音识别没有贡献。

MFCC 是在 FBank 的基础上进行的，所以 MFCC 的计算量更大，且特征相关性小，抵抗噪声的鲁棒性低，一般用于 GMM 的训练。而 FBank 特征保留了更多原始特征，其相邻的特征高度相关（相邻滤波器组有重叠），适用于 DNN 建模，目前基于深度学习模型的语音识别算法一般多采用 FBank 特征。

3.1.2 声学模型

声学模型可以理解为是对声音信号产生过程的建模，它能够把语音输入转换成声学表示的输出，更准确的说是给出语音属于某个声学符号的概率。

1.HMM

HMM 模型，也叫做隐马尔科夫模型，是一种经典的机器学习序列模型，描述为一个隐藏的马尔科夫链随机生成不可观测的隐状态序列，每个状态生成一个观测，而由此产生一个观测序列。

使用 HMM 对声学模型进行建模时，首先会将语音信号根据发音字典编码成三音素序列，然后穷举当前三音素序列所有状态转移序列，转移概率使用状态转移矩阵进行表示，同时使用发射矩阵对音素到文本 token 的发射概率进行表示，通常使用 GMM 或 DNN 对发射概率进行建模，最终使用维特比动态规划算法计算出概率最大的那条状态序列，从而解码出文本结果。

在基于 DNN-HMM 架构的语音识别声学模型中，训练 DNN 通常需要帧对齐标签。而在 GMM 中，这个对齐操作是通过 EM 算法不断迭代完成的，再加上其自身马尔科夫假设这一限制，对真实发声这一物理过程的建模表达能力有限，现在逐渐转向基于深度学习的端到端的语音识别系统的研究。

2.CTC

CTC 的全称是 Connectionist Temporal Classification，就是连接时序分类。它要达到的目标就是直接将语音和相应的文字对应起来，实现编码器输出特征向量和输出标签的对齐，同时再接入一个全连接神经网络预测字符 token 的概率分布，实现时序问题的分类。

其数学模型为:

$$p(w \vee x)=\prod_{t=1}^{T} y_{w_t}^{t}, \forall w \in L^{'}$$

其中 w 表示 token 字符序列，x 表示特征编码器的输出。因为很多帧对应同一个 token，同时很多帧没有任何输出，所以引入一个空白输入 blank 标签，相当于定义了一个多对一的函数，并在输出结果中，把重复的 token 合并，并把空白标签去除，得到最终的识别 token 序列。

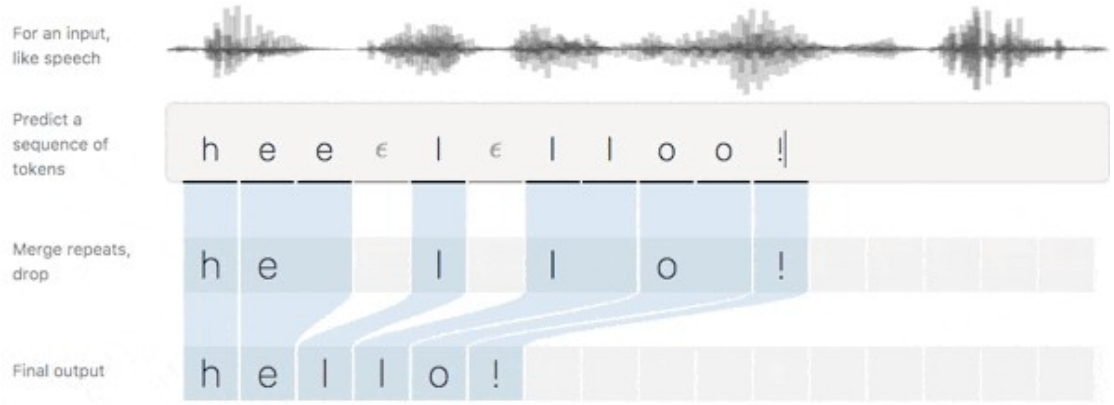


图 3.3 为 CTC 识别效果示意图。

图 3.3 CTC 识别效果示意图

在实际的语音识别系统的使用过程中，CTC 一般用来作为模型训练过程中编码输出和对应标签之间损失函数的计算。首先穷举出所有编码输出帧对应的预测结果 token 路径，并在这些路径中找到所有能通过 CTC 算法映射成标签序列的路径，然后逐一计算生成这些路径的概率，最终训练模型使这些路径的概率之和最大化。从这个算法过程可以看出，需要计算的路径数据会很大，时间复杂度最坏能达到$O(n|t)$，其中 n 代表字典大小，t 代表序列长度，这是系统所不能接受的。所以我们一般使用动态规划算法降低时间复杂度，快速计算损失，其主要思路就是若多种路径使用相同步长映射为相同输出，那么就把这些路径概率合并计算。

图 3.4 为 CTC 损失计算示意图。

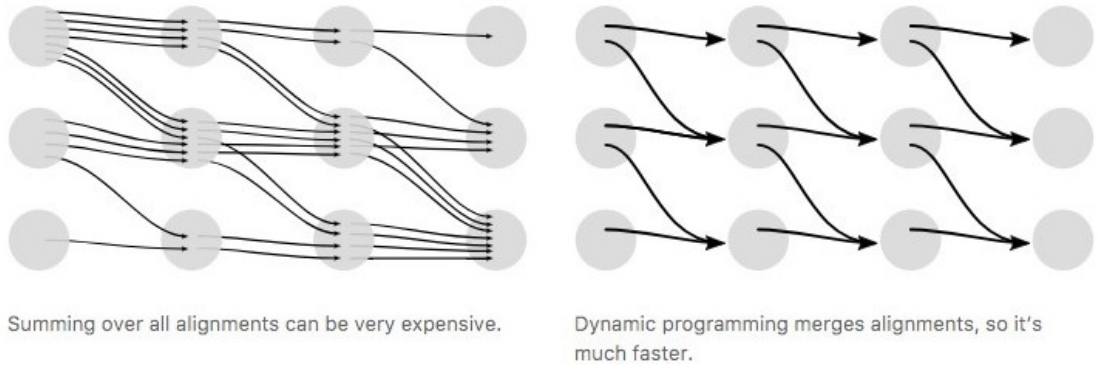


图 3.4 CTC 损失计算示意图

3.LAS

LAS 为基于深度学习 Transformer 思路的一种端到端的语音识别声学模型，其中 L 为 Listen 代表模型的 Encoder 模块，S 为 Spell 代表模型的 Decoder，A 代表 Attention，表示模型使用了注意力机制。

Listen 模块即模型的 encoder 编码器模块，这个模块以由特征提取模块提取的特征向量序列(如 MFCC)作为输入，输出一串相同长度的编码向量序列。因为输入序列中单位时间内就有非常多个向量，且根据发声的连续性原理，相邻的向量信息重合度高，导致训练效率低，所以 编码器通常会先对输入序列作降采样操作，然后通过基于堆叠 CNN，RNN，MLP 等网络 block 进行编码操作，同时加入一些和残差连接等操作提高网络训练效率。编码器的主要作用就是把不同的发声者的发出的相同词语句子的差异和语音数据中的噪声等移除，同时提取语音中与具体内容相关的信息。

Attention 模块则将 encoder 的输出向量与关键字向量 z 0进行 attention 注意力提取操作，计算当前输出所各个输入的权重参数，从而训练模型自动确定当前输出与哪些输入有关，提高模型的表达能力。

Spell 模块即模型的 decoder 解码器模块，此模块使用前一刻和 attention 模块

进行注意力的计算，然后通过一个全连接网络得到一个长度为输出词表大小的向量，代表当前时间步预测输出的分布，重复此操作，即可得到完整的预测序列。

由于神经网络强大的建模能力，相对于 CTC，端到端模型的输出标签不需要进行细分对齐操作，这使模型优化训练变得简单灵活，但是从解码器的计算过程可以看出，必须编码器对所有输入进行一次性编码，解码器才能按时间步进行解码输出，其训练的时间成本较高，且只能应用于离线场景，在需要实时识别的场景使用中一般需要逻辑优化。

3.1.3 语言模型

语言模型就是从大量的文本预料库中，统计出语言文本的一般性语法语义规律，并将这种规律使用某个模型进行表示。对于一种语言的一条文本序列，可以通过训练好的语言模型进行先验概率计算，得到的概率分值即代表了其语言合理性，概率分值越高，表示这条文本序列越符合此种语言的语法语义规律。

将语言模型应用到语音识别算法中，可以引入此种语言的先验知识，很大程度上对直接的声学模型的输出进行错误的语法语义纠正，从而提高整体的语音识别准确率。

常见的语言模型有基于文本统计的 n-gram 语言模型和基于深度学习预训练的 bert 模型，两种模型在模型大小，性能和效果上各有特点，需要根据具体的使用场景进行选择。

3.2 Wenet 语音识别框架

Wenet 是出门问问语音团队和西工大语音实验室联合开发并开源的一款面实际业务场景应用的语音识别框架工具包，主要实现了一种叫做 U2 的两阶段语音识别解码算法，其最重要的特点就是在同一个端到端模型下实现了流式识别和非流式识别两种识别模式，极大扩展了模型服务的应

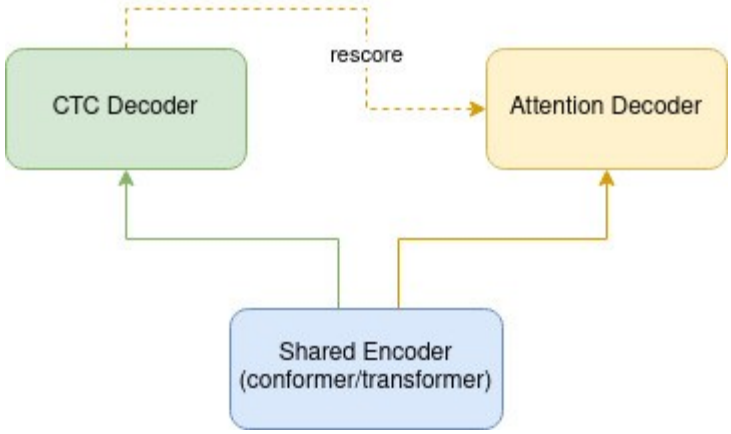
用场景，缩小了语音识别算法的研发和实际工业场景应用之间的鸿沟，这也是本文选择使用 Wenet 作为自动字幕模块的基本语音识别框架的最重要的原因。

近几年来，随着深度学习技术的成熟，基于端到端的语音识别系统在语音识别任务标准词错率（WER）上已经完全超过了 传统的混合语音识别框架，但是在实际的使用场景中，部署一个端到端系统还有度多现实的问题需要解决。

而 Wenet 语音识别框架的两大优势就是能提供流式识别和模型轻量级易于部署。首先 Wenet 使用基于 block 的 conformer 网络结构作为编码器，支持流式输入，decoder 模块根据系统检测到的序列端点进行适时的重打分操作，这样即满足了流式识别的实时性要求，又保证了系统有较高的识别准确率。另外，相对于其他优秀的语音识别开源框架，比如 Espnet，Deepspeech 等，Wenet 的核心设计目标就是高性能易于部署，其没有对很多序列化识别任务进行统一抽象，而是完全聚焦于语音识别任务，同时不依赖于像 Kaldi 等大体量的传统语音识别工具包，完全基于 Pytorch 及其生态系统开发，代码简洁，安装方便，极大提高了开发者的应用部署效率。

3.2.1 模型网络结构

Wenet 语音识别框架实现了 U2 算法，如下图 3.5 所示，其模型结构包含三个网络部分，分别是共享编码器（Shared Encoder），CTC 解码器（CTC Decoder）和注意力解码器（Attention Decoder），其中共享编码器有多个 Transformer 或者 Conformer 网络层组成，其主要作用就是对特征提取的原始特征向量进行信息编码，通过自注意力机制提



取高层抽象特征，为了保持良好的流式识别时延，会设定超参数在注意力计算时使用有限右侧信息，CTC 解码器包含一个线性全连接层，提供编码输出到 CTC 激活映射的转换，对于注意力解码器，包含多个 Transfromer 解码器层

图 3.5 Wenet 网络模型结构示意图

1.流式输入和非流式输入的统一结构

Wenet 针对工业届应用的具体场景需求，兼顾流式响应速度和识别效果，采用了一套两阶段解码的方案，用一个统一的模型结构同时有效的支持流式输入和非流式输入场景。

CTC 解码器进行第一遍流式解码，根据解码分值排序取前 n 个序列作为候选结果，其中分值最高的结果序列可作为流式识别结果实时返回。

多个候选结果再通过注意力解码器做一遍 teacher forcing rescoring 操作，根据得分重新排序。得到更好的识别结果。

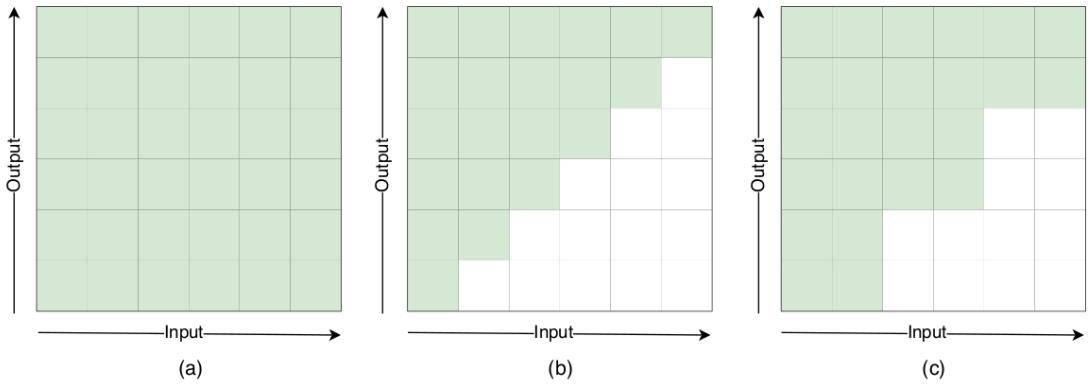
2.对编码器 Conformer 结构进行流式改进

标准的 Conformer 网络结构，self-attention 层需要对整个输入序列进行 attention 得分计算，不能支持流式输入，并且卷积层的计算也需要依赖右侧固定长度的序列信息，依赖的长度也会随着模型层数的增加而增加。所以，需要对这些具体的网络结构进行改进，

使其仅仅依赖有限长度的右侧上下文信息，并且支持流式推理。

对于 self-attention 层， Wenet 选择使用了 chunk 方案进行改进，主要将序列分成多个 chunk，每帧在 chunk 内部做 attention ，这种方案可以保证对右侧依赖的长度与网络层数无关。在基于 chunk 的方案中，chunk 的大小会影响着流式最大延时和模型识别率性能，大的 chunk 延时大但性能更好，小的 chunk 延时小但性能会变差。Wenet 使用了一种新的动态 chunk 训练算法，可以使得模型的 chunk 大小动态可变，在运行时，可根据当前场景延时要求和识别率需求手动调整 chunk 大小，而无需重新训练模型。 而对于 convolution 层，Wenet 使用一个左侧卷积，以支持流式推断。

下图 3.6 为 conformer 网络结构改进图，其中 a 是全序列 attention，b 是左侧



attention，c 是左侧+chunk 内部分 attention。

图 3.6 conformer 网络结构改进图

3.2.2 预训练模型

Wenet 作为一个完善的基于端到端模型的语音识别框架，其提供了很多开源语音识别数据集的预训练模型，没有充足计算资源的研究开发人员提供了一个通用的语音识别能力，也为开发者根据业务需求使用领域数据集进行微调训练提供基础，这也是本文选择 Wenet 的又一重要原因。

表 3.1 为 Wenet 基于不同数据集预训练模型的 CER（字错率），CER 是语音识别任务的常用评价指标，计算公式为：

$$CER=\frac{S+D+I}{N}$$

其中 S（substitution） 表示替换的字符数目，D（deletion）表示删除的字符数目，I（insertion）表示插入的字符数目，N 表示参考序列中字符总数，可以看出，CER 越小，代表语音识别效果越好。

测试集为智慧教室中的课程视频，标签字幕文本已由人工标注。

表 3.1 Wenet 预训练模型测试结果

Datasets	Languages	Testset CER
aishell	CN	10.57
aishell2	CN	9.83
multi_cn	CN	7.54
wenetspeech	CN	7.32

根据在业务数据上的测试对比，基于 wenetspeech 数据集的预训练模型的字错率最低。WenetSpeech 含有 10000+小时的高质量标注数据，覆盖各种互联网音视频、噪声背景条件、讲话方式，来源领域包括有声书、解说、纪录片、电视剧、访谈、新闻、朗读、演讲、综艺和其他等 10 大场景，而非专业的录音棚录制数据，数据质量更符合工业实际应用场景，所以本文选择此数据集预训练的模型作为本系统的初始训练模型。

3.2.3 解码算法

Wenet 在训练时同时使用了帧级别 deocder 的 CTC loss 和 label 级别 decoder 的 loss。因此在识别推理时可以使用多种不同的解码方式进行灵活组合解码。

下图 3.7 为 Wenet 解码流程示意图。

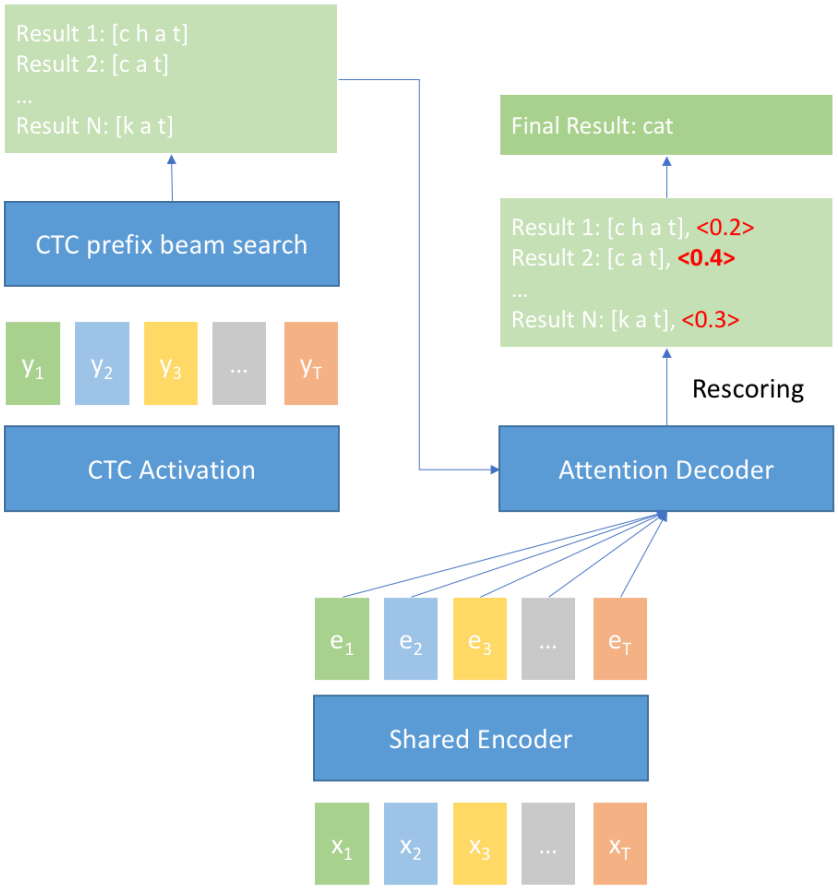


图 3.7 Wenet 解码流程示意图

目前 Wenet 支持四种解码算法

1.CTC greedy beam search， 帧级别解码，解码过程不合并前缀，每一个时间步都选择概率最大的作为结果，最终进行 ctc 序列处理。此解码算法本质上使用贪心思路，得到的结果一般不是最优解，但是优点是解码搜索速度快，效率高。

2.CTC prefix beam search ， 帧级别的解码，每一个时间步合并相同的 ctc 序列前缀求和计算，选择概率和最高的 n 个前缀序列作为当前候选集合，并重复此计算过程，最终得到 n 个最优解码结果。其本质是一种动态规划算法，保证得到最优解的前提下提高计算效率，也是语音识别领域中最常用的解码方式。下图 3.8 为 CTC prefix beam search 解码流程示意图。

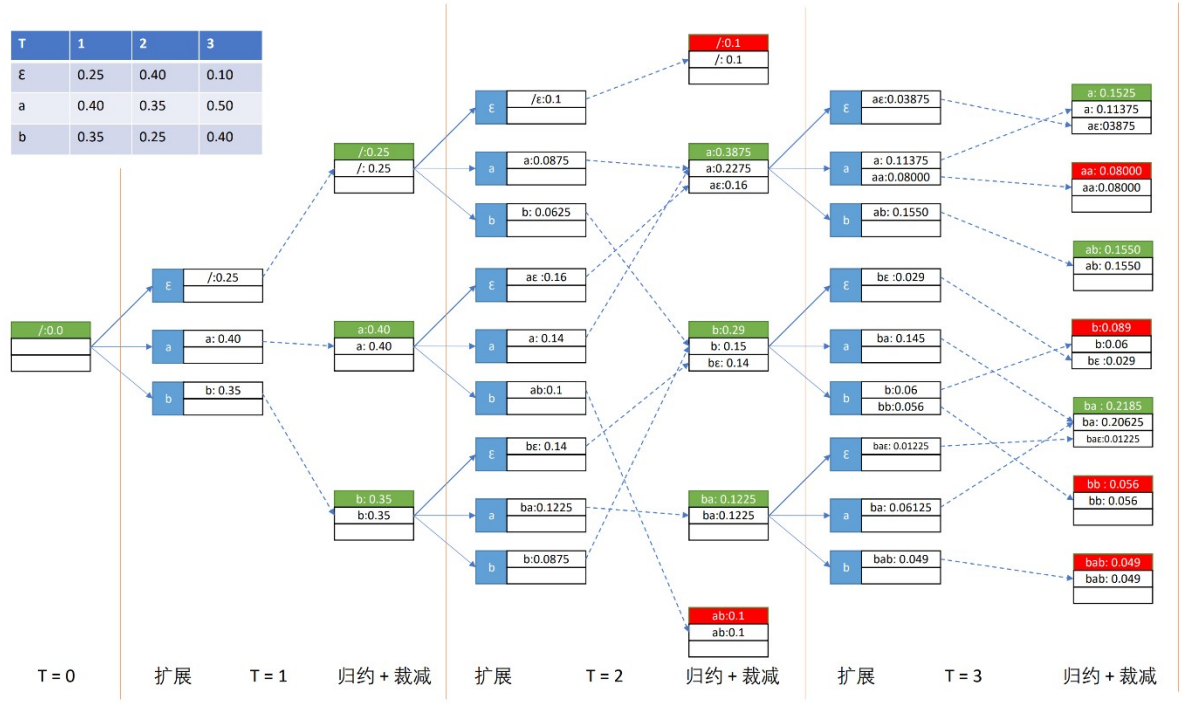


图 3.8 CTC prefix beam search 解码流程示意图

Prefix beam search 的过程，每个时刻有三个动作

（1） 扩展：根据前缀串和当前时刻的输出，计算新串的概率。

（2） 规约：将规约串相同的候选概率相加。

（3） 裁剪：仅仅保留 top k 个最好的序列做下一时刻的拓展，绿色的表示保留，红色的表示被裁减掉，图示中 k 为 3。

3.Attention decoder beam search，label 级别解码， 基于 cross-attention 的注意力解码器单独解码，不支持流式识别。

4.CTC + attention rescoring ， 将 CTC decoder 的 n-best 结果，通过 attention decoder 进行重打分。

CTC + attention rescoring 方案可以利用 CTC decoder 的输出作为流式结果，在损失很小准确率的情况下支持场景的流式需求，并最终利用 rescoring 改进结果，得到一个识别率更高的最终结果。这也是本系统在模型推理实现和应用时采用的解码方案。

3.3 领域数据集微调训练

已有预训练模型是已经拥有了较为通用的语音识别能力，但是对于具体的领域内的业务需求还是效果不足， 所以在此基础上进行微调训练，使其更好的适应领域内的业务和数据。

3.3.1 数据集简介

对于文本来说，业务需求就是教师在线授课场景下的语音内容识别，所以本文微调训练使用的数据是好未来公司开源的 TAL_ASR 语音识别数据集，TAL 数据集是好未来线上课程的教师授课音频标注数据，和本文的使用场景非常接近，且涵盖语文，数学两门学科，为后续基于不同学科定向优化识别效果提供了基础。此数据集一共有 80 多为授课教师内容，内容时长一共 112 小时，每条音频只有一位说话教师，标注数据包含了授课科目，说话人编号及音频内容对应的语言文本，同时也对数据集进行了训练集，验证集和测试集的划分，划分比例为 7：1：2。

3.3.2 训练过程

1.数据集整理

将原始数据集中音频文件和标注文本结果进行对应，生成模型框架训练要求的输入数据格式文件，文件格式为 json 字典格式，三个键值内容分别是说话人编号，wav 音频文件绝对路径和标注文本。下图 3.8 为模型训练数据格式示意图：


```
{"key": "129540", "wav": "/home/wav/train/S0015/129540.wav", "txt": "嚯对吧来一块算一算啊哎你这样算就好了对吧应该是呃二十加三十再加五十是多少"}
{"key": "129541", "wav": "/home/wav/train/S0015/129541.wav", "txt": "一百对不对那么再算一算四十加六十是多少"}
{"key": "129542", "wav": "/home/wav/train/S0015/129542.wav", "txt": "一百那一百加一百呢"}
{"key": "129543", "wav": "/home/wav/train/S0015/129543.wav", "txt": "不要拦着我我要去买买买买买点啥呢同学们你们说说你们想要啥我就买什么想要什么"}
{"key": "129544", "wav": "/home/wav/train/S0015/129544.wav", "txt": "二百对吧所以应该是二百二百别忘了还有谁呀"}
{"key": "129545", "wav": "/home/wav/train/S0015/129545.wav", "txt": "还有这里二百一十为什么是二百一十呢"}
{"key": "129546", "wav": "/home/wav/train/S0015/129546.wav", "txt": "什么哦这里是吗"}
{"key": "129547", "wav": "/home/wav/train/S0015/129547.wav", "txt": "呃稍等我看一眼啊"}
{"key": "129548", "wav": "/home/wav/train/S0015/129548.wav", "txt": "我抄错了啊哎呀"}
{"key": "129549", "wav": "/home/wav/train/S0015/129549.wav", "txt": "我说呢我说我没算错呀原来"}
{"key": "129550", "wav": "/home/wav/train/S0015/129550.wav", "txt": "对不对我说你们为什么要喊你们要喊对吧哦刚才就有人提醒我是吧"}
{"key": "129551", "wav": "/home/wav/train/S0015/129551.wav", "txt": "啊好的好的那下次我错了啊直接打叉叉"}
{"key": "129785", "wav": "/home/wav/train/S0015/129785.wav", "txt": "叫做观察"}
{"key": "129786", "wav": "/home/wav/train/S0015/129786.wav", "txt": "观察什么呢观察这道题目好算不好算"}
{"key": "129787", "wav": "/home/wav/train/S0015/129787.wav", "txt": "哎什么叫好算呢好算应该具备的特点有两个"}
{"key": "129788", "wav": "/home/wav/train/S0015/129788.wav", "txt": "要么数非常的小比如说一加一等于"}
{"key": "129789", "wav": "/home/wav/train/S0015/129789.wav", "txt": "啊很好算对吧要么数非常非常的"}
{"key": "129790", "wav": "/home/wav/train/S0015/129790.wav", "txt": "整比如说一万加一万等于"}
{"key": "129791", "wav": "/home/wav/train/S0015/129791.wav", "txt": "哎可是就怕什么呢"}
{"key": "129792", "wav": "/home/wav/train/S0015/129792.wav", "txt": "大家看一看这个字念什么"}
```


图 3.8 模型训练数据格式示意图

2.提取 cmvn 特征

cmvn 是一个数据归一化统计量，也被称为倒谱均值方差归一化，到谱系数就是本文用到了语音信号提取的声学特征，在预训练模型的数据中已经统计了这个归一化基准值，但是微调训练的数据集的这个基准值会不同，所以需要重新计算，得到全局归一化统计基准，共模型数据归一化处理使用，消除所有音频信号的一些共有属性噪声，也能加快模型训练的收敛速度。

3.统一标签字典

由于预训练和微调训练使用不同的数据集，所以其统计的文本字典也会不同，在原始预训练数据集的统计字典基础上，使用新的数据集统计字典进行补充，特别注意在补充的过程中，不能

更改 blank，unk，sos/eos 等特殊 token 的编码值。

4.模型训练

模型配置参数使用预训练时的配置参数，其他训练参数如下表 3.2 所示：

表 3.2 微调训练参数表

机器设备	Nvidia GTX 1060 GPU
batch_size	8
max_epoch	20
lr	0.002
optim	adam
scheduler	warmuplr
accum_grad	4
use_cmvn	True

在训练过程中，通过日志模块记录了 tal 数据集上的 train loss 和 val loss，同时为了对比参照也记录了模型在 wenetspeech(预训练模型数据集)上的 val loss，各个 loss 的训练变化过程如图 3.9。

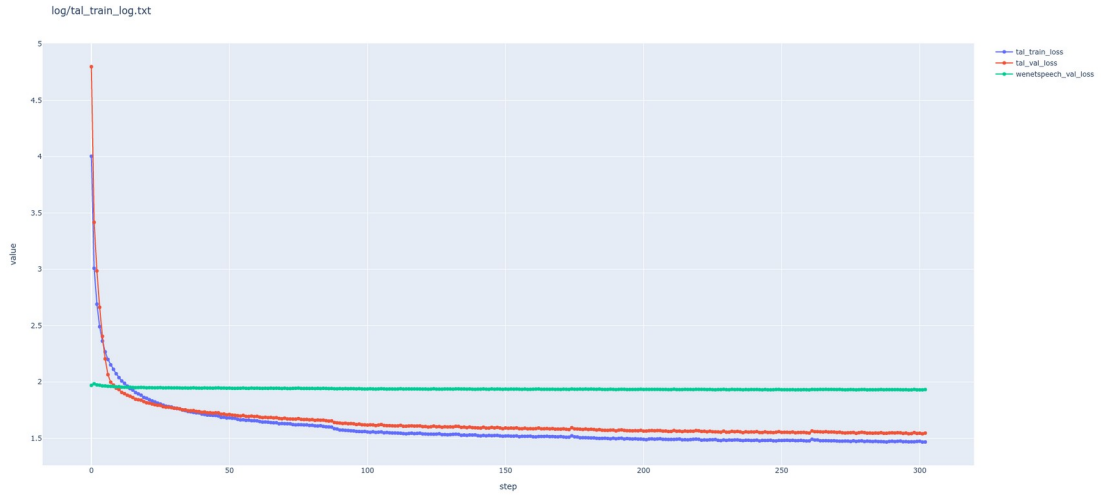


图 3.9 模型训练损失图

由图 3.9 可以观察出，在训练过程中 train loss 和 val loss 都平稳降低，在训练 15 个 epoch 之后，val loss 不再降低，为了防止过拟合，提前停止训练，最终的 val loss 高于 train loss，符合预期逻辑。另外，wenetspeech 的 val loss 先升高，后降低，最后趋于平稳，证明经过在领域数据集上的微调训练对模型在通用语言识别上的能力基本没有影响，同时还能提高在专业领域数据内的识别效果。

3.3.3 微调训练测试结果

微调训练后，为了验证训练效果，分别用预训练模型和微调训练后的模型对 TAL 测试集进行测试，测试效果如下表 3.3:

表 3.3 微调训练对比结果

	解码算法	CER
预训练模型	CTC prefix beam search	9.73
	+attention rescore	9.26
微调模型	CTC prefix beam search	8.54
	+attention rescore	8.37

从测试结果可以看出，经过微调训练后，模型在教育领域测试集上的语音识别效果由稳定提升。

3.4 语言模型

根据上述章节 3.1.3 可知，语言模型对语音识别系统至关重要，加入语言模型可以很大程度提高识别准确率，但同时也应注意到，语言模型的引入也会提高系统计算量，增大系统识别延迟，所以本文在引入语言模型的同时，使用一些逻辑算法策略，在合适的时机进行语言模型打分计算，兼顾系统识别准确率和效率。

图 3.10 为系统解码流程图，其中解码算法有两个分支，一条是使用标准的 CTC prefix beam search 对 CTC 激活输出进行解码，另一条是加入语言模型的解码搜索算法，在 prefix beam search 的基础上，使用语言模型进行筛选过滤候选路径，重复更新搜索，最终输出 N-best 结果。

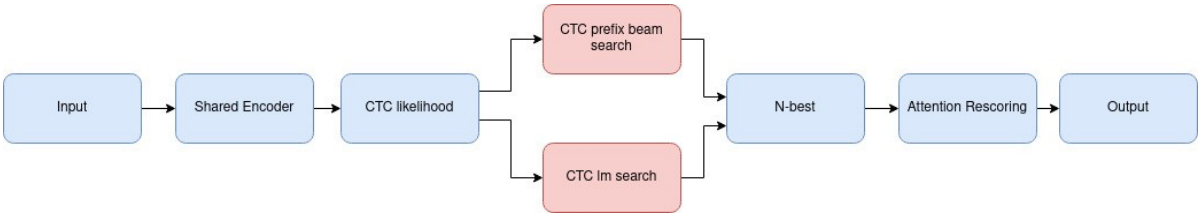


图 3.10 语音识别系统解码流程图

3.4.1 基于 Kenlm 的 n-gram 语言模型

n-gram 是一种基于统计的传统语言模型，其主要思想就是将语言文本序列按照大小为 n 的

滑动窗口进行操作，每次截取一个长度 n 的文本片段序列，文本序列中的每一个 token 叫做 gram,在语料库中，统计所有 gram 出现的频率以及所有截取的 ngram 出现的频率，从而能够计算出截取片段在当前 gram 为前缀的条件下出现的概率，对于一个完整的句子，可以把所有的条件概率相乘得到句子出现的概率，并将其作为评价句子合理性的评分指标。

对于一个 m 个词组成的文本序列，我们希望计算其出现概率，即句子的合理性概率，根据条件概率链式法则，同时加入马尔科夫假设，得到如下计算公式:

$$p(w_1, w_2, \ldots, w_m) = p(w_i \mid w_{i-n+1, \ldots, w_{i-1}})$$

其中 wi 为序列中的第 i 个词。

每一个词 w 都具有 v（词典大小）种取值，这样构造的理论词对数量非常多，但根据语言规则，实际的语料中不会有这么多词对组合，根据最大似然估计准则，得到的概率大部分都为 0，这就为连乘计算带来麻烦，为了解决这个问题，会引入各种数据平滑算法，包括加一平滑，回退平滑等。

n-gram 由于计算简单，部署灵活，效率较高，但是由于其使用了马尔科夫假设，不能考虑长远的上下文语义，所以一般只能用来实现一条句子的语法通顺性的打分操作。

本文使用 Kenlm 来实现 n-gram 打分功能，Kenlm 是一个轻量级的语言模型工具包，并对计算过程进行了很多工程和算法优化，推理速度快，可以高效完成对大量候选序列的语法合理性评价筛选工作。

3.4.2 基于 bert 预训练语言模型

bert 是一个预训练的语言表征模型，由于其内部的自注意力机制和在大量无监督语料上进行了预训练，能更好的关联上下文语义和句子的整体情感信息，所以具有出色语言表征能力。

bert 采用遮蔽语言模型 MLM 进行自回归训练，就是随机 mask 掉句子中 15%的内容，根据前后已知的词来预测 mask 的内容，相当于完形填空，可使模型学习词与词的关联。所以我们计算 bert MLM 层输出的似然分布概率，其代表了当前词在当前句子的上下文语境下出现的合理性，然后把所有的概率值相乘，其结果作为 bert 语言模型对句子的评分值，最终使用此评分结果对 ngram 语言模型的粗筛结果进行精筛操作，得到语法语义都更为合理的候选集。

3.4.3 语言模型搜索处理逻辑

本文中语言模型的引入采用候选序列重打分机制，包括 ngram 粗筛和 bert 精筛两个步骤，最后对筛选结果中的序列通过语言模型得分和 CTC 似然得分的加权分值进行排序，获得最终的 n-best 识别结果。

图 3.11 为语音识别系统语言模型处理流程图。

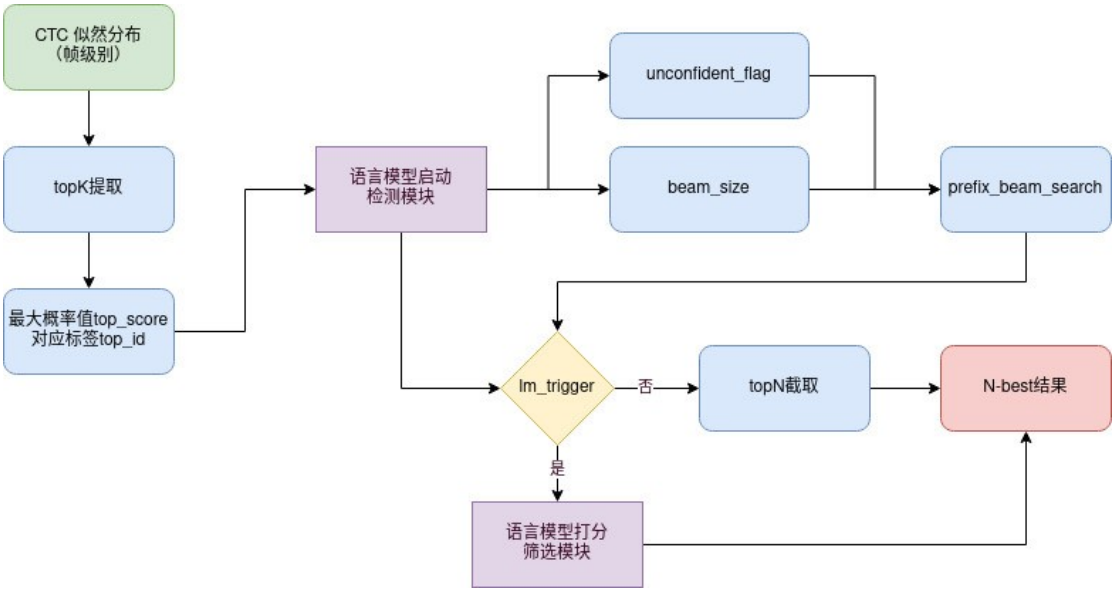


图 3.11 语音识别系统语言模型处理流程图

首先读取帧级别的 CTC 似然概率分布值，然后对每一帧根据概率值排序，获取前 k（k 为系统超参数）大的标签，然后经过语言模型启动检测模块进行逐帧检测，若未检测到语言模型启动标志，则进行普通的 prefix beam search 进行搜索更新，但同时将预测结果不确信的帧进行冗余搜索，相反，若启动标志为真，则调用语言模型打分筛选模块对冗余候选结果进行筛选，得到最终的 N-best 结果。

1.语言模型启动检测模块

主要功能就是对 CTC 的预测输出进行检测判断，从而确定启动语言模型进行纠正筛选的合理时机，在确保语言模型打分准确的前提下，同时满足系统的实时识别的需求。

图 3.12 为语言模型启动检测模块程序流程图。

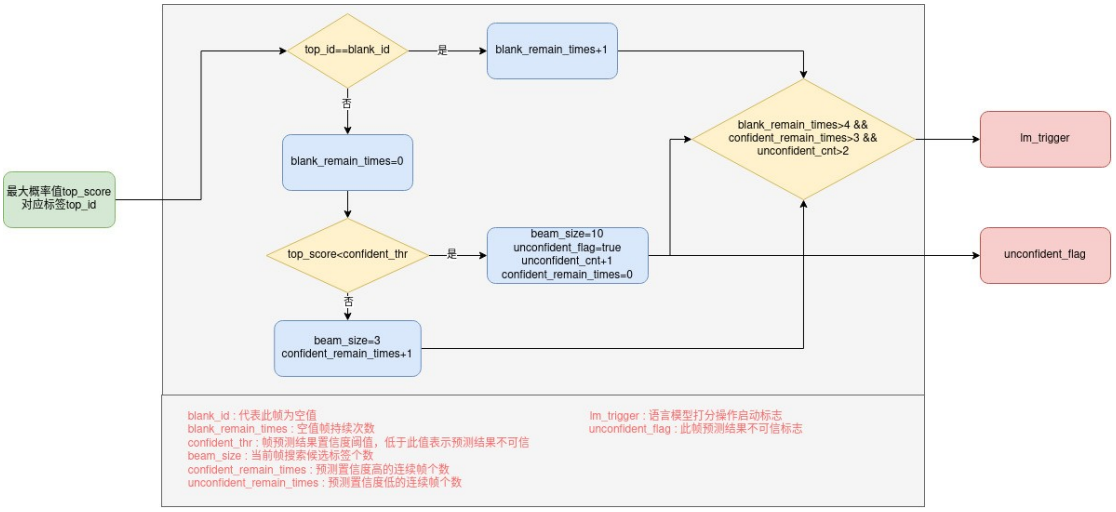


图 3.12 语言模型启动检测模块程序流程图

其基本思路就是设定一个 CTC 预测输出的置信度阈值，若某帧的最大概率低于此阈值，则认为此帧的 CTC 输出不可信，同时统计空白帧持续次数，可信帧持续次数和不可信帧一共出现的总次数，若当前序列中出现超过 2 个不可信帧同时当前帧为空白帧状态时，则将语言模型启动标志置为真。并且，对于每一帧，都会将当前帧的置信状态进行输出，若当前帧为不可信帧，则在 prefix beam search 过程中将此帧的 top10 似然概率值的标签都加入到搜索序列中，防止正确结果漏出，为之后的语言模型筛选提供冗余候选集。

2.语言模型打分筛选模块

此模块的主要功能就是为候选序列集合进行语言模型评分，从而筛选出语法语义都更合理的序列结果。由 3.3.1 和 3.3.2 章节的研究可知，n-gram 的效率高，但效果一般，而 bert 模型的效果较好，但是时间性能较差，表 3.2 展示了两种语言模型的时间性能对比结果，所以本文将两种模型联合使用，设计了两层筛选机制。

表 3.4 语言模型打分效率

	Time(s)	Score
n-gram	0.000572	0.079
bert	0.013	0.149
测试序列	那年冬天祖母死了父亲的也差事差也交卸了	

图 3.13 为语言模型打分筛选模块处理流程示意图

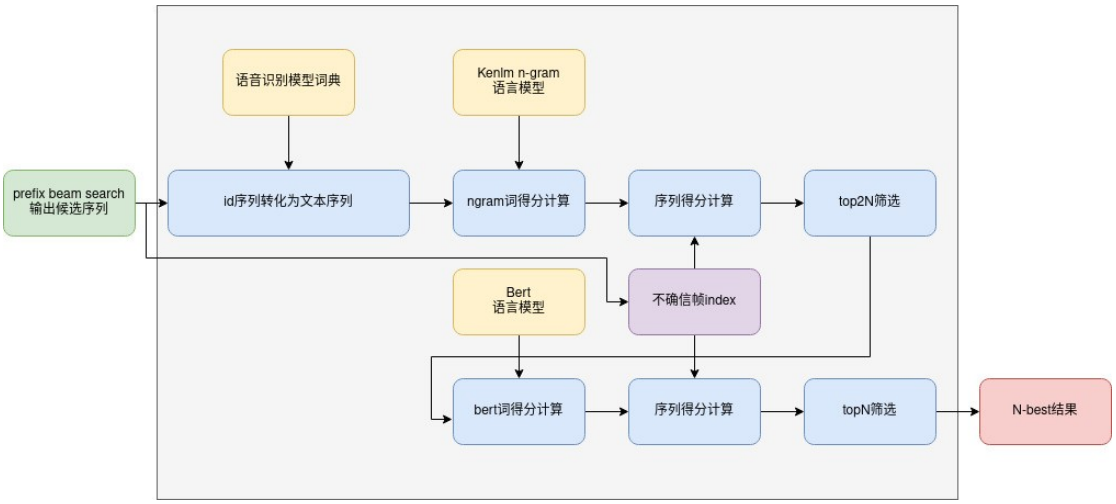


图 3.13 语言模型打分筛选模块处理流程示意图

首先会对候选 ID 序列基于语音识别模型词典转化为文本序列，因为考虑语言模型设计的独立性，语言模型和语音识别模型一般不采用同一个标签词典，所以转化为统一的文本序列后便于语言模型处理。

之后对所有的候选序列经过 ngram 语言模型进行打分操作，为了不受其他其他确信帧的语言模型得分值的影响，根据输入的不确信帧的序号统计不确信帧的得分之和从而得到完整序列的得分，得到 2n 个候选结果。然后调用 Bert 语言模型重复此操作，从而筛选出 n 个最终候选集。

3.两层语言模型筛选实例

下图 3.14 为一条测试样例序列的 CTC 预测输出结果。其中红色柱状图中的 token 序列是 CTC 的预测结果，横坐标下的为真实的标签序列。

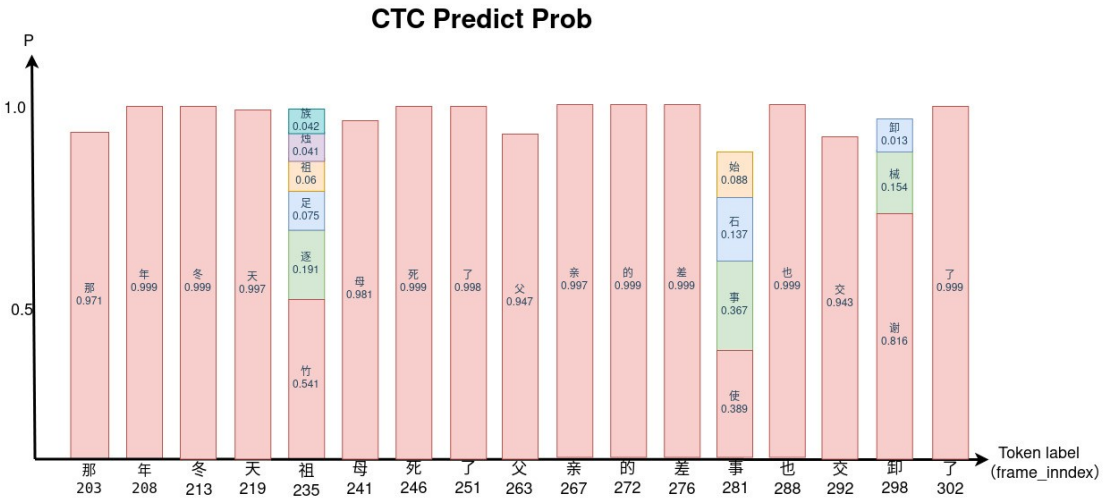


图 3.14 CTC 预测输出样例图

根据语言模型检测模块的识别检测，第 235，281，198 帧为不确信帧，且不确信帧数大于最小阈值，所以启动语言模型筛选，不考虑 CTC 规则中的重复合并情况，一共有 $6\times4\times3=72$条候选序列组合，将所有候选序列通过语言模型进行打分操作，并根据分值进行排序截取，得到最终的 N-best 结果。

下图 3.15 是语言模型打分排序结果，可以看出，正确的序列组合分值最高，即是通过语言模型筛选得到正确预测结果序列。

```
2023-01-17 19:02:52.485 | DEBUG    | pycorrector.detector:_initialize_detector:89 - Loaded language model: /home/sfy/.pycorrector/datasets/zh_giga.no_cna_cmn.prune01244.klm
00 那年冬天祖母死了父亲的差事也交卸了 0.1073
01 那年冬天祖母死了父亲的差事也交谢了 0.0979
02 那年冬天祖母死了父亲的差事也交械了 0.0944
03 那年冬天足母死了父亲的差事也交卸了 0.0922
04 那年冬天族母死了父亲的差事也交卸了 0.0914
05 那年冬天祖母死了父亲的差使也交卸了 0.0910
06 那年冬天竹母死了父亲的差事也交卸了 0.0907
07 那年冬天逐母死了父亲的差事也交卸了 0.0904
08 那年冬天烛母死了父亲的差事也交卸了 0.0884
09 那年冬天足母死了父亲的差事也交谢了 0.0841
10 那年冬天族母死了父亲的差事也交谢了 0.0833
11 那年冬天祖母死了父亲的差使也交谢了 0.0830
12 那年冬天竹母死了父亲的差事也交谢了 0.0827
13 那年冬天逐母死了父亲的差事也交谢了 0.0825
14 那年冬天足母死了父亲的差事也交械了 0.0811
15 那年冬天烛母死了父亲的差事也交谢了 0.0806
16 那年冬天祖母死了父亲的差始也交卸了 0.0804
17 那年冬天族母死了父亲的差事也交械了 0.0804
18 那年冬天祖母死了父亲的差石也交卸了 0.0803
19 那年冬天祖母死了父亲的差使也交械了 0.0800
20 那年冬天竹母死了父亲的差事也交械了 0.0798
21 那年冬天逐母死了父亲的差事也交械了 0.0795
22 那年冬天足母死了父亲的差使也交卸了 0.0782
23 那年冬天烛母死了父亲的差事也交械了 0.0777
24 那年冬天族母死了父亲的差使也交卸了 0.0775
25 那年冬天竹母死了父亲的差使也交卸了 0.0769
26 那年冬天逐母死了父亲的差使也交卸了 0.0767
27 那年冬天烛母死了父亲的差使也交卸了 0.0749
28 那年冬天祖母死了父亲的差始也交谢了 0.0733
29 那年冬天祖母死了父亲的差石也交谢了 0.0732
30 那年冬天足母死了父亲的差使也交谢了 0.0713
```

图 3.15 语言模型打分排序结果

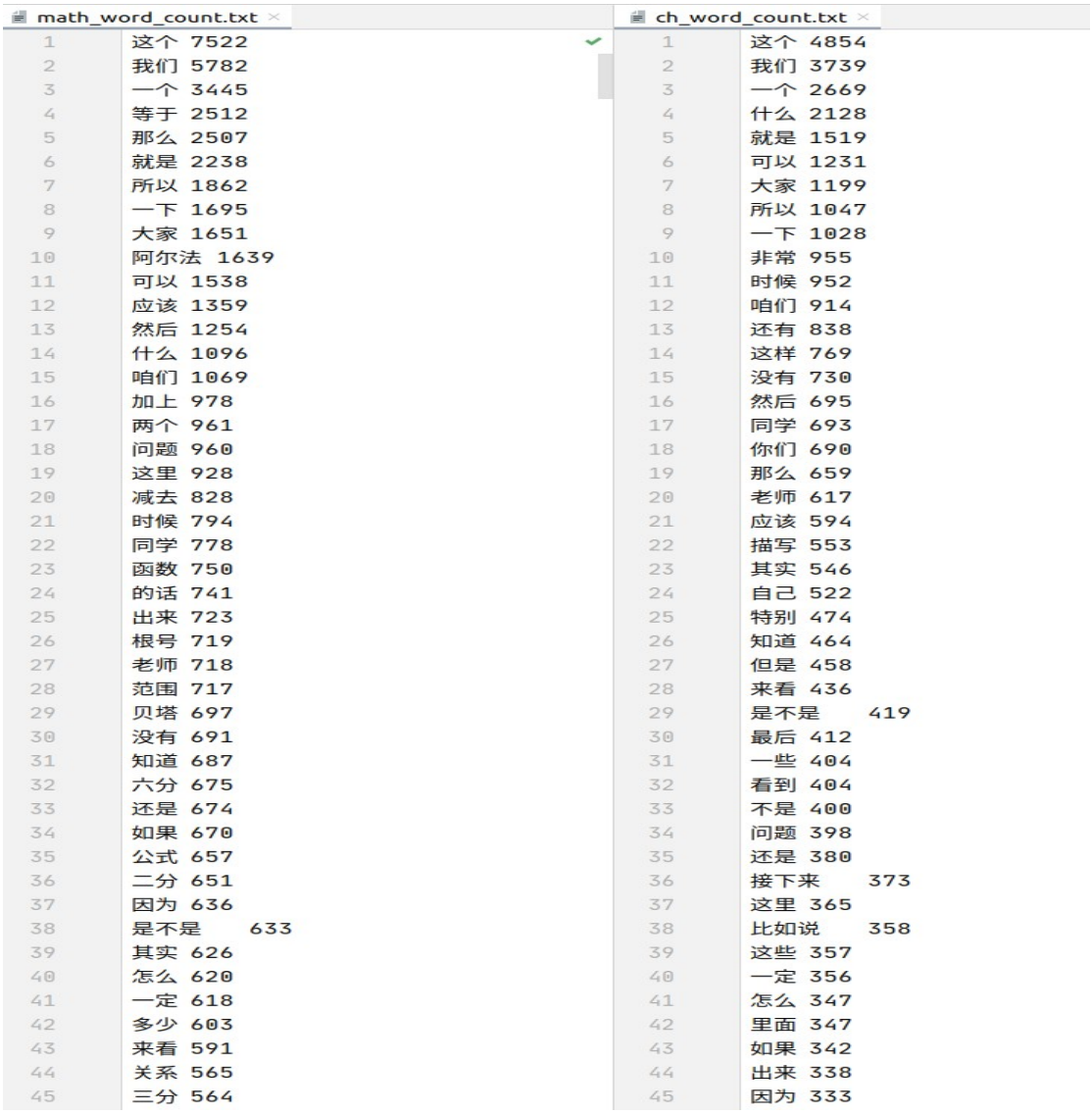
3.5 热词增强模型

虽然在上述章节中讨论了在语音识别系统中加入语言模型可以提高语音识别准确率，但是这些语言模型都是通用语言模型，体量庞大，训练复杂。本文研发的语音识别系统主要是用在教育课堂领域，对于不同的学科，会有不同的学科领域常用词语，所以文本设计了一个热词增强模块，在语音识别打分时引入学科常用词的先验知识，进一步提高实际教学使用场景的识别准确率，从而提高字幕生成效果。

3.5.1 学科常用词统计

基于智慧教室项目，我们收集了不同学科教师授课内容的语料数据，在整理清洗后，通过使用 jieba 分词工具进行分词和词频统计操作，最终整理出学科领域的常用词词典，为后续热词打分模型提供基础。同时在实际实现的过程中，语音识别服务模块会向请求端提供一个学科分类的超参数，由请求端设定当前调用的学科类别，并在识别时调用相应学科的常用词词典进行热词增强操作。

图 3.16 是本文统计的数学学科和语文学科的课堂授课内容常用词词典。



math_word_count.txt

```
这个 7522
我们 5782
一个 3445
等于 2512
那么 2507
就是 2238
所以 1862
一下 1695
大家 1651
阿尔法 1639
可以 1538
应该 1359
然后 1254
什么 1096
咱们 1069
加上 978
两个 961
问题 960
这里 928
减去 828
时候 794
同学 778
函数 750
的话 741
出来 723
根号 719
老师 718
范围 717
贝塔 697
没有 691
知道 687
六分 675
还是 674
如果 670
公式 657
二分 651
因为 636
是不是 633
其实 626
怎么 620
一定 618
多少 603
来看 591
关系 565
三分 564
```

ch_word_count.txt

```
这个 4854
我们 3739
一个 2669
什么 2128
就是 1519
可以 1231
大家 1199
所以 1047
一下 1028
非常 955
时候 952
咱们 914
还有 838
这样 769
没有 730
然后 695
同学 693
你们 690
那么 659
老师 617
应该 594
描写 553
其实 546
自己 522
特别 474
知道 464
但是 458
来看 436
是不是 419
最后 412
一些 404
看到 404
不是 400
问题 398
还是 380
接下来 373
这里 365
比如说 358
这些 357
一定 356
怎么 347
里面 347
如果 342
出来 338
因为 333
```

图 3.16 分学科统计常用词词典

3.5.2 热词模型

在 Wenet 的解码过程中有两种解码算法，分别是 CTC prefix beam search 和 CTC lm search，在按照时间步解码搜索时，这两种算法都会同时保留多条候选搜索路径，然后在融合 CTC 评分和语言模型评分进行综合筛选。而热词增强模型会在搜索过程中维护一个上下文状态图，也叫 Context Graph，这个状态图记录了每条搜索路径中和热词前缀匹配的当前状态，若检测到热词匹配成功，则会根据当前匹配热词的统计词频对当前路径进行相应权重的加分操作，若匹配失败，则进行回退操作，清除 Context Graph 中记录的前缀匹配状态。

声学模型得分和热词增强得分的加权计算公式为：

$$y^{\wedge}=argmax\log P(y \mid x)+\lambda \log P_{C}(y)$$

其中 x 为输入语音信号，y 识别候选序列，c 为当前匹配热词，λ为热词增强得分权重，是一个超参数，由用户调试设定，设定的越大，表示用户越信任包含常用词的候选序列。

下面用一个实际样例来描述热词增强模型实际的工作过程。

假设当前热词词典中有三个领域常用词，分别是 语言，自动字幕，字符串，设定每个字的增强得分为 3，不考虑词频权重，则系统构建的 Context Graph 状态图如下图 3.17 所示：

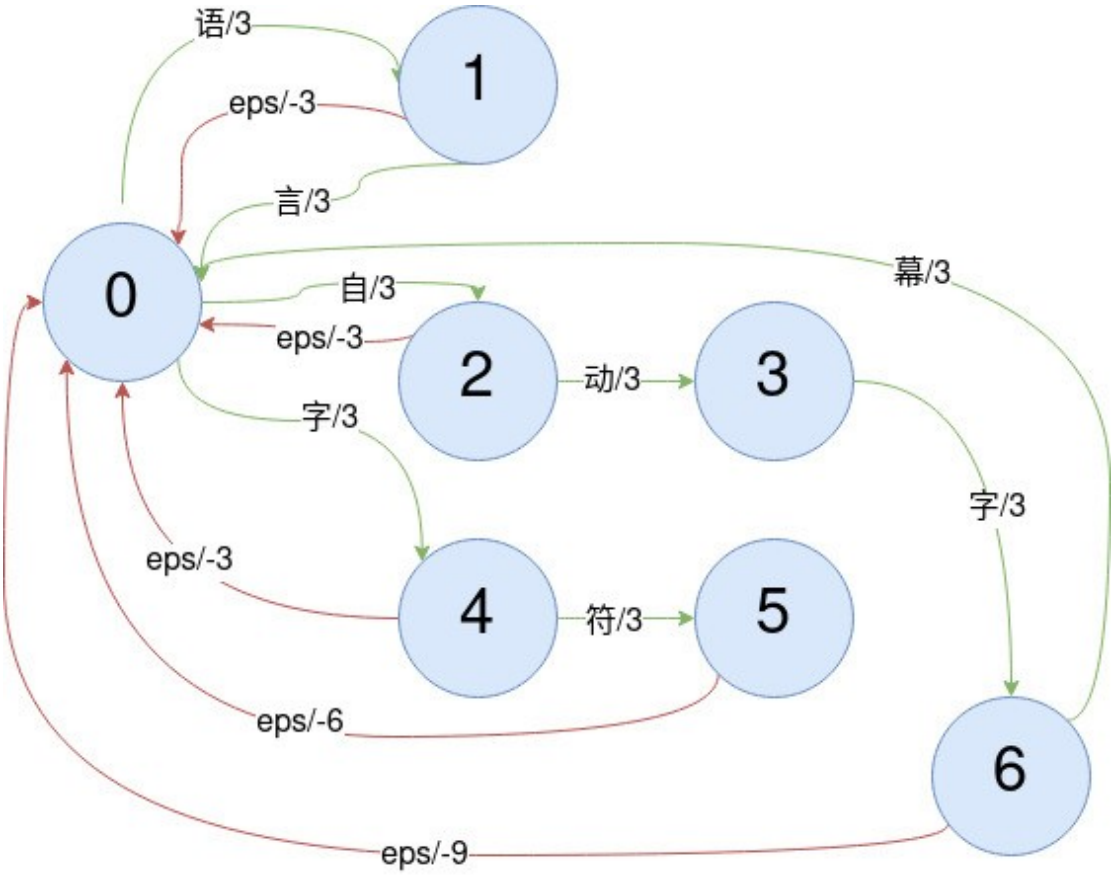


图 3.17 热词状态图

在实际的解码过程中，当某一条路径匹配到相应前缀时，系统会给其相应的分数奖励，这里固定设成 3 分，为了防止某些词只是前缀相同，但是不能完全匹配的情况，所以加了一条回退边 eps，并通过这条边清除之前前缀匹配奖励得到的分数。例如“自动化”最终不能得到完全匹配，所以在搜索到“化”时，清除前面“自动”两个字得到的增强分数。而当候选序列中出现“自动字幕”这个词语时，则可以完全匹配到 Context Graph 中的一个子图结构，则可以获得 12 分的一次热词得分奖励。

3.5.3 实验效果对比

本文在好未来 Tal 数据集上进行对比测试，首先按照学科分别构建学科常用词词典，然后分别从语文和数学两个学科中随机抽取 100 条数据进行测试。

测试字错率（CER）结果如下表 3.4 所示。

表 3.3 热词模型实验结果对比

学科/得分权重	0	1	3	5	10
语文	12.53%	12.16%	11.78%	9.83%	11.32%
数学	15.36%	13.67%	12.04%	10.98%	10.57%

从实验结果可以看出，随着热词增强得分权重的增大，两个学科的字错率都有显著降低。对于语文学科，得分权重超过 5 时，测试集上的 CER 反而开始升高，根据分析由于语文的预料数据和通用预料较相近，当持续增大热词得分权重时，就会出现在常用词上过拟合的现象，在通用语言预料上的效果就会有所下降。而对于数学学科，预料中包含的特殊数字组合和专用数学语言较多，所以对于用通用语料训练的语音识别模型识别效果相对较差，把这些专用数学词汇抽取出来当作热词增强，随着得分权重增加，这些专词汇也会得到很好的识别，整体识别 CER 也得到了稳定的降低，效果明显。

3.6 端点检测

端点检测技术是本文语音识别系统的重要组成部分，相对于上文提到的语音分割技术，其主要是在语音信号级别上的分割，通过去除空白信号的方式降低系统负载，而端点检测则是通过一系列逻辑规则对识别结果序列进行分割，从而在识别端点处结束一个轮次编码解码搜索流程，开始新一轮的识别，最终达到让语音识别模块支持实时长语音的流式识别的效果。

3.6.1 时间戳提取

在 CTC 解码搜索过程中，系统一般会记录各条搜索路径的前缀以及对应得分，根据 CTC 的规则，会在规约的过程中对空白 token 进行去除， 但是去除空白 token 会影响对最终序列结果中每个 token 对应时间戳的计算，所以本文在 CTC 搜索的同时，也会保留记录 CTC 输出的原始 token(包括空白 token)，然后根据语音信号采样率和编码器的降采样率计算出每个时间步的时长，再通过全局搜索时间步数标志得到当前搜索轮次的基准时间，最后计算得到所有前缀中每个 token 对应时间戳，并加入到最终的 N-best 输出中，方便后续模块处理使用。

通常一个字的时间戳信息应该包括起始时间和终止时间，而使用上述算法，我们只能获取该字峰值所在的时间。因此在我们的实现中，考虑到延迟等因素，我们将峰值所在的时间当做该字的终止时间，上一个字峰值所在的时间当做起始时间。

为了实现时间戳提取算法，会记录额外信息，增减系统运行时内存开销，所以本文会定时启动维特比所法得到概率最大的路径时间戳进行减枝操作，减少内存开销。

3.6.2 端点检测规则

基于解码搜索结果中的时间戳信息，可以计算出当前有效解码文字的持续时间以及空白信号的持续时间，根据这些实时信息，同时考虑业务实际需要，本文制定了三条端点检测规则，分别

是:

1.识别出有效文字之前，检测到了持续 5s 的静音;

2.识别出有效文字之后，检测到了持续 1s 的静音;

3.本轮次持续解码超过了 20s 后，检测到持续 500ms 的静音。

3.7 本章小结

本章第一节详细介绍了语音识别技术的基本数学原理，以及在实现过程中的各个处理流程步骤，是实现本文功能的技术基础。

第二节介绍了本系统实现使用到的 Wenet 开源框架，并系统分析了此框架的优缺点和选择此框架的业务需求及理由。

第三节介绍了本系统在教育领域数据集上进行微调训练的过程，以及在实际业务场中取得的效果提升。

第四节介绍了语言模型的使用对语音识别效果的巨大作用，同时详细介绍了本文在系统实现过程中加入语言模型的具体方式和效果。

第五节介绍了一种热词增强技术，通过引入领域常用词的先验知识，进一步增强模型的实际识别效果。

第六节简单介绍了语音识别系统的两个支持性功能，包括时间戳输出和端点检测，为支持长语音识别和后续处理模块提供基础支撑。

。

参考文献[1]

参考文献集中著录于正文之后，不得分章节著录。属于外文文献的，直接使用外文著录，不必译成中文。

“参考文献”四个字与章标题格式相同。参考文献表的正文部分用五号字，汉字用宋体，英文用 Times New Roman 体，行距采用固定值 16 磅，段前空 3 磅，段后空 0 磅，标点符号用半角符号。

参考文献的著录方法和文献的标注方式有关，可采用“顺序编码制”和“著者-出版年制”。“顺序编码制”是指正文中索引文献时，用顺序编号的方法标注文献。文献序号放“[ ]”内，以上标方式标注在索引位置。“著者-出版年制”是指索引文献处用文献著者和出版年度标注文献，一般著者和出版年度放“（）”内，以逗号分隔，标注在索引位置。

[1] 全文参考文献索引方式只能选用“顺序编码制”或“著者—出版年制”其中之一，文献列表也应选择相对应的著录方法，此处作为示例列举了两种方式，实际撰写论文时不得混用。

以“顺序编码制”索引文献时，其参考文献应按索引对应编号顺序著录。以“著者-出版年制”索引文献时，参考文献应按文种分类著录，按著者字母顺序排序，中文文献放前方。

全文参考文献索引方式只能选用“顺序编码制”或“著者—出版年制”其中之一，文献列表也应选择相对应的著录方法，撰写论文时不得混用。

参考文献的具体著录方法和标注方法见论文写作指南附录 A。

附录 A 附录示例

附录是与论文内容密切相关、但编入正文会影响整篇论文的条理性和逻辑性的一些资料，是论文主体的补充项目，并不是必须的。以下内容可置于附录之内：

a. 放在正文内过分冗长的公式推导；

b. 方便他人阅读所需要的辅助性教学工具或表格；

c. 重复性数据和图表；

d. 非常必要的程序说明和程序全文；

e. 关键调查问卷或方案等。

附录的格式与正文相同，并依顺序用大写字母 A，B，C，……编序号，如附录 A，附录 B，附录 C，……。只有一个附录时也要编序号，即附录 A。每个附录应有标题。附录序号与附录标题之间空一个汉字符。例如：“附录 A 参考文献著录规则及注意事项”。

附录中的图、表、数学表达式、参考文献等另行编序号，与正文分开，一律用阿拉伯数字编码，但在数码前冠以附录的序号，例如“图 A.1”，“表 B.2”，“式（C-3）”等。

致谢

本论文是在 xx 老师的悉心指导下完成的。xx 老师作为一名优秀的、经验丰富的教师，具有丰富的 xx 知识和 xx 经验，在整个论文实验和论文写作过程中，对我进行了耐心的指导和帮助，提出严格要求，引导我不断开阔思路，为我答疑解惑，鼓励我大胆创新，使我在这一段宝贵的时光中，既增长了知识、开阔了视野、锻炼了心态，又培养了良好的实验习惯和科研精神。在此，我向我的指导老师表示最诚挚的谢意！

……

(仅为网络示例，可根据论文实际进行撰写，使用时把模板示例内容尽皆删除即可)

正文内容：采用小四号宋体，两端对齐书写，段落首行左缩进 **2** 个汉字符。行距为固定值 **20** 磅，段前空 **0** 磅，段后空 **0 磅**

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明：所呈交的学位论文，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，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：　　日期：　　年　月　日

学位论文使用授权说明

（必须装订在提交学校图书馆的印刷本）

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

- 按照学校要求提交学位论文的印刷本和电子版本；
- 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务；

- 学校可以采用影印、缩印、数字化或其它复制手段保存论文；
- 因某种特殊原因需要延迟发布学位论文电子版，授权学校□一年/□两年/□三年以后，在校园网上全文发布。

（保密论文在解密后遵守此规定）

论文作者签名：　　导师签名：

日期：　　年　月　日