**1 项目主要研究**

1.1 了解研究背景与研究现状

声纹识别技术的原理，声纹识别技术的发展历程，声纹识别技术的现状

针对学习和调研的结果，完成综述的撰写

1.2 搭建系统的基本框架

设计带安全验证的声纹识别系统的逻辑流程，设计简单的GUI界面

基于pyqt5的gui界面，可以展示运行流程

1.3 口型识别模块--定位

捕捉视频文件中每帧的人脸区域，在此基础上通过颜色与移动特征定位嘴唇区域

完成了固定帧率下视频文件的逐帧捕捉，并加以方形框进行标识，进而导出为新的视频

**2 项目研究成果**

嘴唇区域定位 源文件

基于skvideo的音频信号特征提取 源文件

**3 项目季度报告**

**3.1. 如何将声纹识别与口型识别相匹配？**

基于常识的推论是，由于人类通常能够将听到的声音与给定的一组嘴唇运动进行匹配，因此嘴唇运动与由语音特征表示的听到的声音高度相关。

然而，目前还缺乏同时融合时空视听信息来解决音频流和视频流是否匹配的根本问题的研究。

AVR系统的方法是利用从一个模态中提取的信息，通过对缺失信息的补充来提高另一模态的识别能力。其根本问题是找到音频流和视频流之间的对应关系

有些学者的初步探索：

一个耦合的三维卷积神经网络（3D-CNN）架构，它可以将两种模式映射到一个表示空间中

三维CNNs同时从空间和时间两个维度提取特征，从而在相邻帧中捕获和连接运动信息

—>设计非线性映射

—>与输入空间相比，具有显著的降维，并且针对区分匹配和非匹配视听流进行了优化

**3.2. 嘴唇特征提取的实现**

OpenCV的接口可以实现人脸定位。

cvHaarDetectObjects是opencv1中的函数，opencv2中人脸检测使用的是 detectMultiScale函数。它可以检测出图片中所有的人脸，并将人脸用vector保存各个人脸的坐标、大小（用矩形表示），函数由分类器对象调用

**3.3. ivector**

因为从UBM模型自适应到每个说话人的GMM模型时，只改变均值，对于权重和协方差不做任何调整，所以说话人的信息大部分都蕴含在GMM的均值里面。GMM均值矢量中，除了绝大部分的说话人信息之外，也包含了信道信息。联合因子分析(Joint Factor Analysis, JFA)可以对说话人差异和信道差异分别建模，从而可以很好的对信道差异进行补偿，提高系统表现。但由于JFA需要大量不同通道的训练语料，获取困难，并且计算复杂，所以难以投入实际使用。由Dehak提出的，基于I-Vector因子分析技术，提出了全新的解决方法。JFA方法是对说话人差异空间以与信道差异空间分别建模，而基于I-Vector的方法是对全局差异进行建模，将其二者作为一个整体进行建模，这样处理放宽了对训练语料的限制，并且计算简单，性能也相当。

**3.4. GMM-UBM**

GMM(高斯混合模型)。它将空间分布的概率密度用多个高斯概率密度函数的加权来拟合，可以平滑的逼近任意形状的概率密度函数，并且是一个易于处理的参数模型，具备对实际数据极强的表征力。但反过来，GMM规模越庞大，表征力越强，其负面效应也会越明显：参数规模也会等比例的膨胀，需要更多的数据来驱动GMM的参数训练才能得到一个更加通用（或称泛化）的GMM模型。

GMM-UBM模型最重要的优势就是通过MAP算法对模型参数进行估计，避免了过拟合的发生，同时我们不必调整目标用户GMM的所有参数（权重，均值，方差）只需要对各个高斯成分的均值参数进行估计，就能实现最好的识别性能。

**3.5. 声纹识别模块的实现**

初步完成了输入训练音频-预处理-训练识别模型，输入验证音频-预处理-验证的框架，其中预处理步骤包括了盲源分离与音频切割以适配周围环境嘈杂的情况，同时考虑到数据量较小时i-vector系统表现优于x-vector系统，训练与识别模型暂时使用MATLAB AUDIO工具包中ivectorSystem及相关函数实现（已修复原函数中的一处错误）。测试时使用了1422段完整音频训练i-vector系统，以5个说话人共10段完整音频作为登记（enroll）数据，验证结果总体正确率达70-80%，基本符合立项时的目标，但仍然存在提升空间，目前暂定计划为将i-vector系统替换为文本相关的d-vector系统。

1. n=[62,65,28,43];
2. s=zeros(1,198);
3. iv=ivectorSystem('SampleRate',fs,'InputType','audio');
4. adsTrain=audioDatastore('D:/source\_mono/\*/\*.wav','FileExtensions','.wav','LabelSource','foldernames')
5. trainLabels = adsTrain.Labels;
6. adsTrain = transform(adsTrain,@(x)extract(afe,x));
7. enrollLabels = adsEnroll.Labels;
8. adsEnroll = transform(adsEnroll,@(x)extract(afe,x));
9. trainExtractor(iv,adsTrain,'UBMNumComponents',64, 'UBMNumIterations',5, 'TVSRank',32, 'TVSNumIterations',3);
10. trainClassifier(iv,adsTrain,trainLabels, 'NumEigenvectors',16, "PLDANumDimensions",16,"PLDANumIterations",5);
11. **for** i=1:710
12. **for** j=1:4
13. dist(j)=sqrt(sum((y(i,:)-c(j,:)).^2));
14. **end**
15. loc=find(dist(:)==min(dist));
16. loc=loc(1);
17. name=cell2mat(ttmp.textdata(i));
18. name=name(21:30);
19. adsTest=audioDatastore(['D:/source\_mono/',name,'/\*.wav'],'FileExtensions','.wav','LabelSource','foldernames');
20. reset(adsTest);
21. sim=zeros(1,198);
22. **for** j=1:numel(adsTest.Files)
23. features=extract(afe,read(adsTest));
24. res=identify(iv,features,'plda');
25. **for** k=1:181
26. **for** p=1:199
27. **if** p<=198 && contains(cell2mat(tttmp.textdata(p)),**string**(res.Label(k)))
28. break;
29. **end**
30. **end**
31. **if** p<=198 && idx(p)==loc
32. sim(p)=sim(p)+**double**(res.Score(k));
33. **end**
34. **end**
35. **end**
36. s(i)=sum(sim)/n(loc)/numel(adsTest.Files);
37. **end**

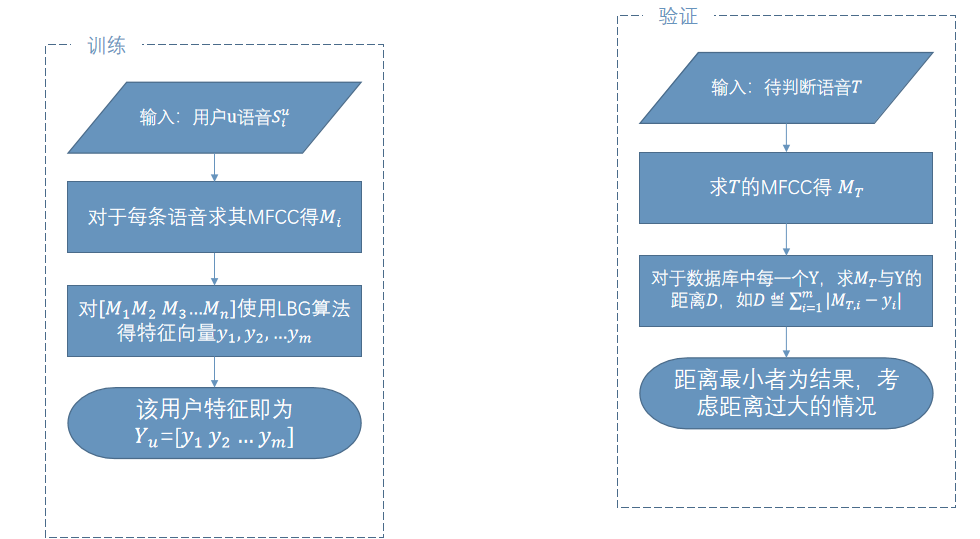


图 声纹识别模块注册-验证框架流程图

完成了从用户注册到验证声纹的框架，声纹识别算法仍采用MATLAB R2021b及Audio Toolbox中ivectorSystem与相关函数实现。i-vector系统下的UBM模型训练数据为144段独立字符音频的Mel频率倒谱系数，用户注册时录入27段音频（即数字2至数字9与数字0各3段）作为该用户的原始数据，验证时随机生成4位数字密码（不含数字1），用户录入验证音频，经过按字符切割、单声道化（取左右声道算术平均值）与提取特征的预处理后，密码中的每种数字，以所有对应该数字的音频特征为登记数据进行声纹识别，识别结果为未注册说话人或识别分数低于预设阈值则保持闭锁，结果均为同一已注册用户且识别分数大于预设阈值则开锁，并将所有录入音频与结果加入数据库中。由于文本相关的方法涉及数据量极小，使用矢量量化算法可能会带来额外的复杂度，故无需对其进行可能丢失信息的简化处理，反而应利用其时间复杂度小的特点，使用动态的登记数据训练模型以增加系统的可靠度。

**3. 6. 3D卷积神经网络的交叉视听识别**

**概况：**

* 当音频损坏时，视听语音识别（Audio-visual recognition，AVR）被认为是完成语音识别任务的另一种解决方案。
* AVR 系统的方法是利用从某种模态中提取的信息，通过填补缺失的信息来提高另一种模态的识别能力。

**关键：**

* 音频和视频流之间的对应关系

**初步实现：**

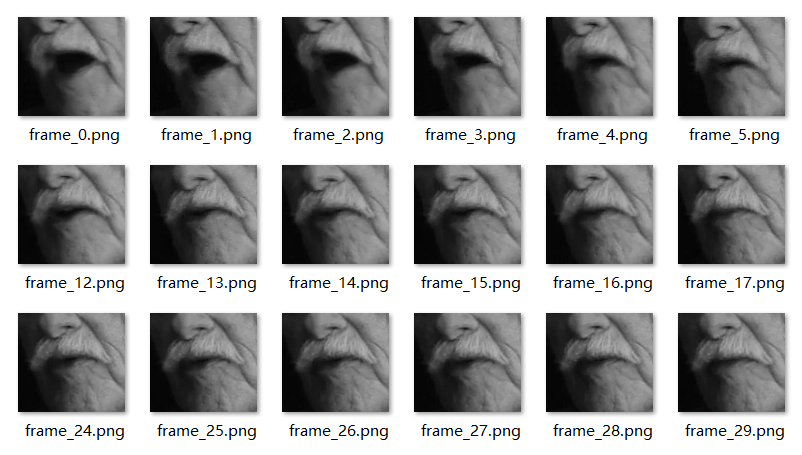
基于dlib opencv-python

shape\_predictor\_68\_face\_landmarks

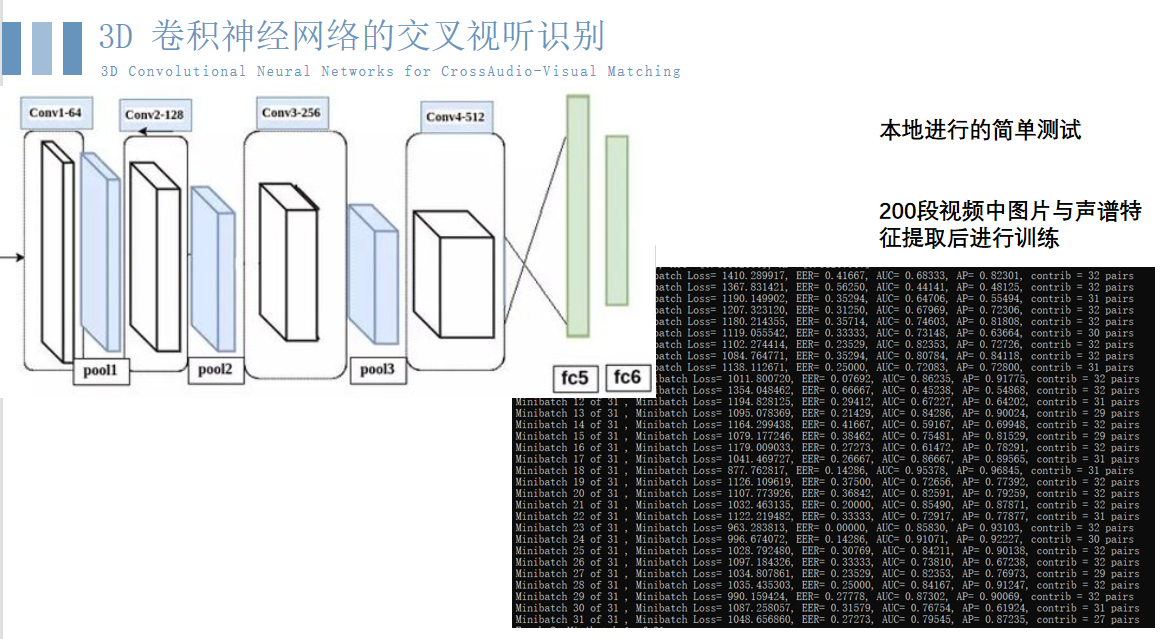
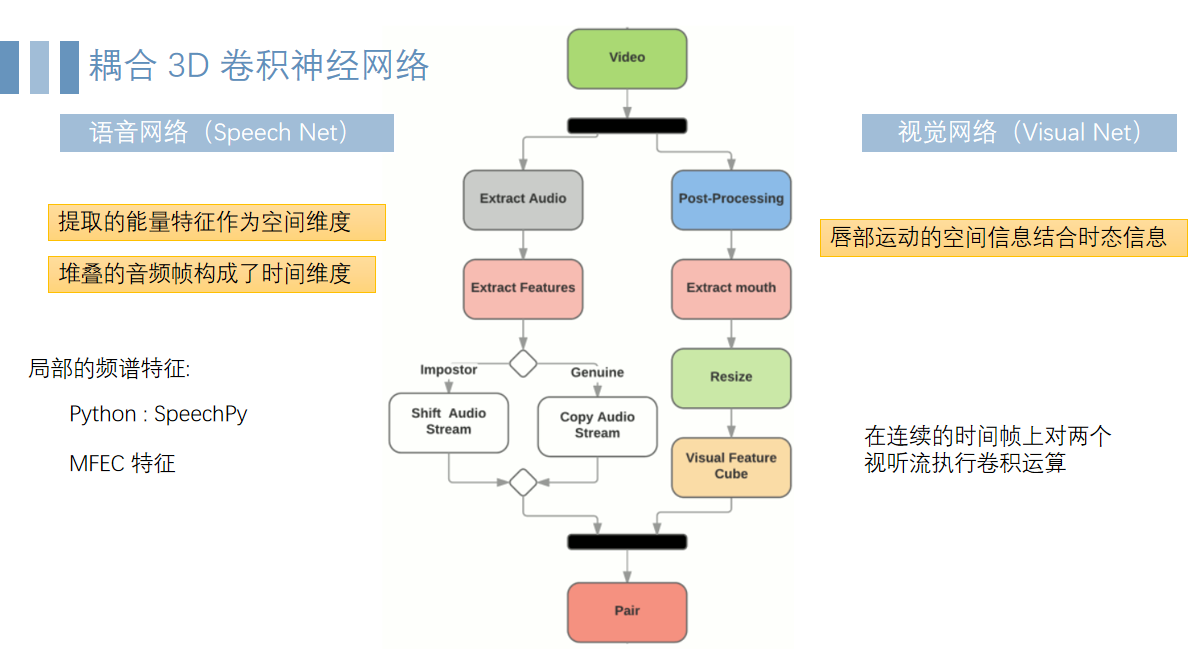
口角 = 48–60

嘴唇分数 = 61–67



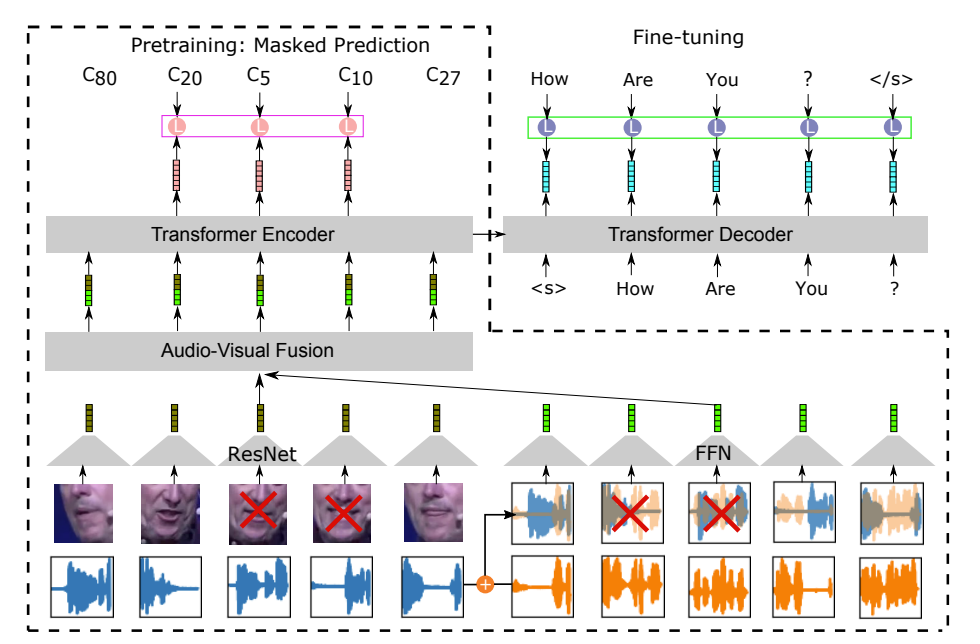


**耦合 3D 卷积神经网络：**



**3.7. 基于av-hubert框架的视听识别系统**

文本相关的说话人识别涉及自然语言处理领域。通过结合人们说话过程中嘴唇和牙齿活动、语音方面的信息，捕捉到音频和视频间的微妙联系。这和人类本身感知语言的模式很相似。通过一个transformer框架可将掩码音频、图像序列编码为视听特征，预测离散的任务序列，捕捉两种模态之间的相关性，尝试通过自监督学习的过程来实现掩蔽聚类的任务。



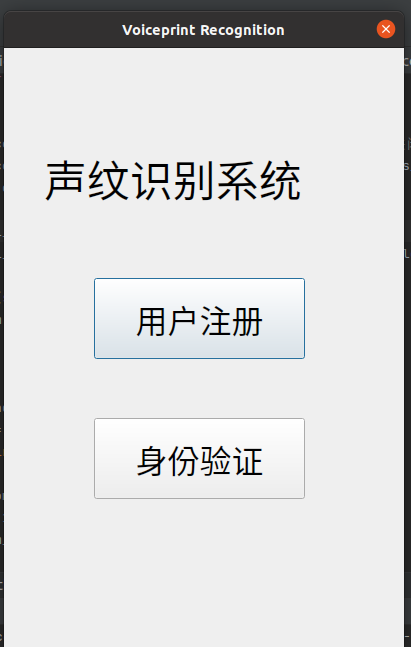
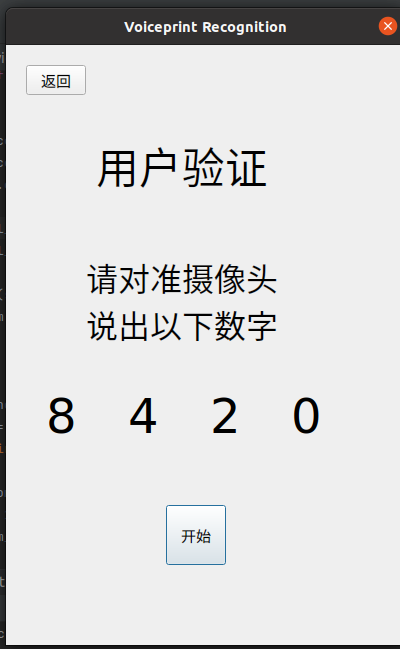
**3.8. 环境部署与系统运行实例**

Linux是一种自由和开放源码的类UNIX操作系统。因为其低成本、强大的定制功能以及良好的移植性，目前Linux已经被移植到许多计算机硬件平台，可以运行在服务器和其他大型平台之上，也广泛应用在嵌入式系统上，如手机等。在移动设备上广泛使用的Android操作系统就是创建在Linux内核之上。

在之前的声纹识别和唇纹验证中，为了编写和调试的方便，我们都选择在Windows平台上进行开发。但是相较于Linux，Windows没有那么强的可移植性，其成本也相对较高。因此，在前期基于Windows平台测试成功的基础上，我们继续将其移植到Linux平台。为此，我们在系统上安装了Ubuntu系统，同时将所需的依赖也移植到系统中。例如，在Ubuntu上安装Anaconda，以便进行Python包管理；并安装MATLAB，以便可在Linux上运行MATLAB的函数。并且在Python环境下安装matlab\_engine，使Python能够调用MATLAB的API，有助于各项工作成果的合并。

目前已基本实现将各项分支功能移植到Linux平台并成功运行，并在Linux平台上使用Python实时录音，对一开始使用Pyqt5编写的GUI界面进行修改，使其与声纹识别，唇纹验证的功能接口相结合，形成了一个具有实际可操作性的系统。

**运行流程：**

1. 进入初始界面，可选择注册用户或验证声纹  
   
2. 在用户注册界面，输入用户名，同时录入三次录音，检查无误后可保存  
   
3. 在用户验证界面，随机生成4个数字，点击开始即可录音，并对声纹和唇纹进行检测

给出识别结果：  
