# 面向智能手机的说话人识别研究

计111（10111939）陈楚楠

摘要：目前说话人识别研究主要是基于专业录音设备和处理机，来提取用户的语音中的特征并训练模型和识别确认处理。主要适用于大型安全领域。整套解决方案的可移动性不强，且不利于大规模普及和部署。因此本课题旨在研究开发出一个面向智能手机的说话人识别系统，在智能手机上获取用户语音，并在手机上提取特征并训练模型和识别确认处理。从而实现说话人识别方案的大规模普及和快速部署。本课题开发的说话人识别系统主要面向Android平台手机用户。

关键词：智能手机，说话人识别，特征提取，训练模型，Android

## 1 研究背景

### 1.1 课题背景

说话人识别最初是应用在司法领域，用于帮助对嫌疑人的查证或判定罪犯，经过不断发展后进入安保和军事领域，用于一些机密场所出入控制，机要设备的使用控制和战场监听等等。

出此之外，说话人识别还有着广阔的市场应用前景。例如在通信和互联网尤其是新兴的移动互联网领域，说话人识别技术可以应用于语音拨号，电话银行，信息服务，安全控制，账户登录，E-mail，即时通信等。

### 1.2 开发背景

#### 1.2.1 说话人识别

说话人识别是一种生物认证技术。通过语音信号中的波形变化反映说话人生理和应为上的特征，并根据特征识别说话人。这些特征涉及到说话人的年龄，性别，感情，种族等等。

说话人识别的大致过程是首先录入说话人的语音样本，提取其中的语音特征并保存以待应用。在识别时将待测试的语音的特征与保存的语音特征做对比，从而确定说话人身份。

根据识别系统对待识别语音内容的不同，又可以分为文本相关和文本无关两种方式。

文本相关的说话人识别方式要求说话人发音的关键词或关键句子与训练文本相同，且在识别时也要按照相同内容发音。

文本无关的说话人识别方式在训练和识别时都不对说话人的的发音内容做要求，其识别对象是任意的语音信号。

由于文本无关的说话人识别方式的无法控制的特性，而文本相关的识别系统的鲁棒性较强，本课题所研究开发的系统采用文本相关的说话人识别方式。

#### 1.2.2 Android平台

本课题开发的说话人识别系统是面向Android平台手机用户。Android是一种基于Linux的开源的自由的操作系统，主要使用于智能手机，平板电脑，相机，游戏机，电视机等移动设备。最初由由Andy Rubin开发，后在 2005年8月被Google收购，随后Google以Apache开源许可证和GPL的授权方式，发布了Android的源代码。

2008年，在GoogleI/O大会上，谷歌提出了Android架构图，在同年9月，谷歌正式发布了Android 1.0系统。2011年第一季度，Android在全球的手机市场份额跃居第一。在2014年第二季度，分析机构Strategy Analytics发布的智能手机操作系统全球情况报告显示，目前Android操作系统的全球市场份额已达84.6%，是有史以来最高比重。

从Android刚刚发布时的1.0版本到目前的5.0版本，Android逐渐从一个崭新的系统发展为最成熟的移动端操作系统。

Android作为市场份额最大的手机端操作系统，得益于其的平台开放性和开发开放性。Google允许任何手机终端厂商加入Android平台联盟，也正因为众多的手机厂商加盟，使得Android平台拥有了众多的开发者，开发的Android应用更加丰富，用户数越来越多。

Google提供给Android一个相当自由的开发环境，有着不停更新的Development Toolkit，和较少约束的系统API，使得开发者可以自由地进行开发。

### 1.3研究的目的与意义

通过本课题的面向Android平台手机用户的说话人识别系统，可以对用户进行生物身份识别确认以替代原本的数字密码验证，避免了数字密码遭冒充者窃取的风险。

除此之外还使得生物身份识别更简单易用，在安保、司法等以外领域的地方部署，可以极大地促进说话人识别技术的发展。

最后，目前移动互联网行业发展迅速，但是安全性还是原地踏步，本系统还可以与Android平台上的其他应用进行绑定，以极大地增强一些移动互联网应用的安全性，例如手机支付钱包等功能。

## 2 文献综述

### 2.1 Android开发

Android开发有4个层次(如图1所示)，自下而上分别是Linux内核层开发，类库层开发，应用框架层开发，应用层开发。

第一层Linux内核层开发由C语言实现，主要负责管理Android硬件的驱动。

第二层类库层开发依然由C语言，主要负责Android上一些常用组件的开发，例如SQLite数据库，WebKit浏览器内核，libc常用的C语言函数库，Dalvik虚拟机。

第三层是Android系统应用框架由Java语言实现，包含了Activity Manager界面管理，Notification Manager通知管理，Resource Manager资源管理等等，实现了对Linux内核层和类库层的封装，让Android不过度依赖与Linux内核，做到内核无关的特性，让Android框架开发和Android应用开发能在不考虑Linux内核和驱动下顺利完成。

第四层是Android应用程序的开发由Java语言实现，可以利用Android提供的API接口，实现各种功能的应用，例如信息服务，拍照，录音，定位等等。

图1

Android应用程序由4大组件构成，分别是Activity、Service、Broadcast Receiver、Content Provider。Activity是Android应用与用户交互的界面，可以显示一些控件也可以监听并处理用户的事件做出响应。Broadcast Receiver用来对外部事件进行过滤，从而只获得自己感兴趣的内容。Service 是没有用户界面的程序，主要用来开发如监控类等后台程序。Content Provider用于给应用提供各种各样的数据，例如SQLite数据库，SharedPreferences配置文件。

本课题的开发的说话人识别系统主要在第二层和第四层完成。先在第二层中编写语音类库，再在第四层中开发一个应用调用这个类库。

### 2.2 隐马尔科夫模型(HMM)

隐马尔可夫模型（Hidden Markov Model，HMM）是一个统计模型。它用来描述一个含有隐含未知参数的状态跳转过程。其要点在于从可观察的参数中找出这一系列状态跳转过程中的隐含参数。然后利用这些参数来作进一步的分析，例如语音识别。

在正常的马尔可夫模型中，状态参数对于观察者来说是完全直接可见的。而在隐马尔可夫模型中，状态参数并不是直接可见的，但受状态影响的某些参数变量则是可见的。

例如在某地的某人在雨天喜欢宅在家中，在晴天时喜欢出去踢球，那么当你得知这个人在某一天出去踢球(或宅在家中)，就可以推断出这个人所在的地方的天气是晴天(或雨天)。这个人告诉了你的他的活动，也就是一个受天气影响的参数变量，而我们对天气并不是可知的，这样一个系统就是隐马尔可夫模型。

在语音识别领域，最常用到的是隐马尔可夫模型的解码问题。

通过给定测试序列(测试用户的语音特征序列) 和模型参数(事先训练好的语音特征序列集合)，再利用Viterbi算法寻找某种意义上最优的状态序列(测试用户的身份)。这就是在语音识别领域使用隐马尔科夫模型的过程。

### 2.3 语音特征提取

语音特征提取是指采用数字技术与模拟技术相结合，选择和提取语音信号中的特征，得到说话人的模型，而不是直接从语音信号中得到。无论是训练时的语音还是测试时的语音，都要通过语音特征提取才能进行后续建模处理。

对于说话人识别，语音信号中提取到的特征需要满足能较好地区别不同用户的能力，又能对相同用户同一。

人们能对不同人的语音做出区别是人们的耳蜗类似于一个滤波器组，对语音信号进行过滤后找出不同说话人的特征。因此想要在计算机上对语音进行特征提取就要模拟出一个类似人耳的滤波器组。

当频率在1000HZ以下时，人们的耳朵的感知能力与频率成线性关系；但当频率在1000HZ以上时，人们的耳朵的感知能力与频率不再构成线性关系，而更偏向于对数关系，这就使得人们的耳朵相比于对比高频信号来说对低频信号更敏感，故采用Mel 频率倒谱系数(即 MFCC)。

Mel 频率倒谱系数提取过程如下：

1. 对语音信号s(n)进行预加重，分帧，加窗等处理，得到每个帧的时域信号x(n)。
2. 将上述对时域信号x(n)后补0，直至形成长度为N的序列(N一般为256)，然后进行快速傅立叶变换(FFT)后得到线性频谱x(k)：
3. 将上一步的线性频谱x(k)通过Mel频谱滤波器，并取每个三角形的滤波器频率带宽内的所有信号幅度加权和作为某个带通滤波器的输出，从而得到Mel频谱，并通过对数能量的处理，得到对数频谱s(m)：

为各个带通滤波器的传递函数。

1. 将上一步的对数频谱s(m)经过离散余弦变换(DCT)得到倒谱频率，即可得到Mel频率倒谱系数c(n)：

## 3 技术路线

### 3.1 功能模块



图2

经过简单的需求分析，本课题开发的说话人识别系统主要有一下几个模块(图2)：

* 锁屏界面模块

通过Android系统的API对Android手机进行锁定，通过AudioRecord类实现对用户的语音的记录，并进行特征提取和解码，当通过系统确认时锁定解除。

* 登陆模块

本课题开发的系统的登录界面，要求用户输入密码，防止非法用户登录本系统进行修改。

* 主界面模块

本课题开发的系统的主界面，是其他功能模块的入口所在。

* 训练模块

系统的训练界面，通过AudioRecord类实现对用户的语音的记录，并进行特征提取和建模训练。

* 测试模块

系统的测试界面，通过AudioRecord类实现对用户的语音的记录，并用来测试训练的数据是否准确。

* 用户管理模块

系统的用户管理界面，通过与数据库的关系映射实现用户的添加删除功能。

* 设置模块

系统的设置界面，用来设置阈值等信息。

### 3.2 Native功能模块

Native功能模块是指由C编写但是运行在Java虚拟机上的功能模块，拥有较好的性能，并且兼容C语言中众多的类库工具包。经过简单的测试，Native功能模块包含两个过程，分别是训练过程和测试过程，主要分为三块(图3)：

* 特征提取模块

对AudioRecord类得到的实例WAV文件进行特征提取，得到MFCC特征文件。

* 训练模块

对若干个MFCC特征文件进行建模训练，得到HMM模型。

* 测试模块

依据已有的HMM模型群，对某个MFCC特征文件进行解码。

图3

### 3.3 MVC模式

Android上应用开发普遍遵循MVC设计模式，MVC全称是Model模型，View视图，Controller控制器，在Android应用上的体现就是Entity实体例如文件、用户，XML界面布局，Activity界面模块。通过MVC设计模式，可以将数据和业务逻辑解耦，将业务逻辑集中部署，在定制数据结构和改进业务逻辑时可以不需要重新编写全部代码。使用MVC设计模式可以提高代码质量，提升开发效率。

本系统的MVC框架设计如图4所示。

图4

### 3.4 SQLite数据库以及ORM框架

SQLite是Android应用中最常用的数据库。使用C语言构建，并集成在Android系统的第二层——类库层中。Android应用可以使用SQLiteOpenHelper类中的方法，传入Context作为参数，得到SQLiteDatabase，在通过execSQL方法将SQL语言传入并执行。

使用SQLiteOpenHelper提供的方法操作数据库并不符合面向对象的软件开发思想。因此本课题在操作数据库的过程中使用了ORM对象关系映射(Object Relational Mapping)框架，目前ORM框架可谓汗牛充栋，但为了能扎实掌握这一门技术以及更好地适应于本课题，在借鉴了一些ORM框架后，改编了其中之一xUtils，使其成为更小更适合本课题的xutil\_orm框架。通过使用ORM框架，不仅可以省去编写SQL语句的困扰，还可以更好地对一些异常和事务进行处理。

本课题中预计主要将用户，文件，设置参数三类结构存放于SQLite数据库中。其中用户表与用户实体的映射如图5所示。



图5

### 3.5 Apache-commons-io类库

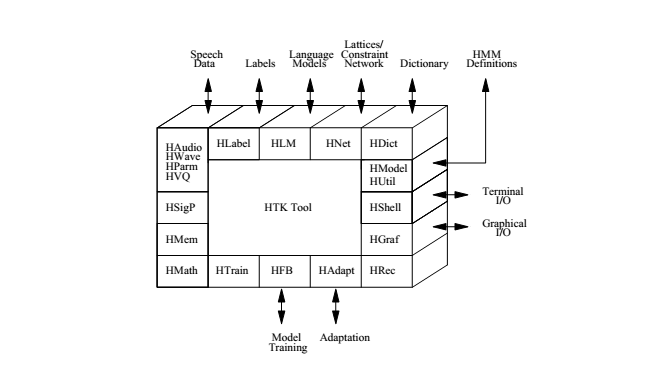
本课题中涉及到不少文件的操作，例如WAV语音文件的新建，MFCC特征文件的新建、复制与删除，HMM模型文件的新建、复制与删除，以及文件路径的传递。为了使得本系统有更好的健壮性，以及遵守软件工程中“不重复造轮子”的思想，本系统中使用了Apache-commons-io类库。通过其中的FileUtils类可以清晰简单地对文件、文件夹进行新建、修改、删除、复制、获取路径等操作。使用Apache-commons-io类库可以提升开发效率，并且更好地处理一些文件异常。

### 3.6 HTK工具箱

HTK工具箱是一个用于构建和操作隐马尔科夫模型的便携式工具箱。HTK主要用于语音识别的研究，虽然已被用于包括语音合成研究字符识别和DNA测序的许多其他应用程序。

HTK工具箱的多数功能已被建立成库模块。这些模块确保了每个工具对外部的接口使用完全相同的方法，并提供了通用功能的中心资源。图6说明了一般的HTK工具的软件架构和输入输出接口。

* HShell控制用户的输入输出以及与操作系统的交互。
* HMem控制所有的内存管理。
* HMath提供数学支持。
* HSigP完成语音分析所需的信号量处理操作。
* HLabel提供标签文件的接口。
* HLM用于语言模型文件。
* HNet用于网络和网格。
* HDict用于字典。
* HVQ用于矢量量化码本。
* HModel用于隐马尔科夫模型（HMM）的定义。
* HWave完成所有在波形阶段的语音的输入输出。
* HParm完成语音的参数化阶段。
* HWave和HLabel支持多文件格式并允许数据从其他系统输入。
* HAudio支持音频的直接输入。
* HGraf提供简单的交互式图形。
* HUtil提供大量用于计算HMM的通用程序。
* HTrain和HFB支持多种HTK训练工具。
* HAdapt提供多种适应工具的支持。
* HRec包含了识别过程中使用的主要函数。

图6

HTK工具的使用如图7所示有四个主要的阶段：数据准备，训练，测试，分析。

1. 数据准备

对于语音信号，可以使用HCopy参数化波形，得到MFCC特征文件。

1. 训练

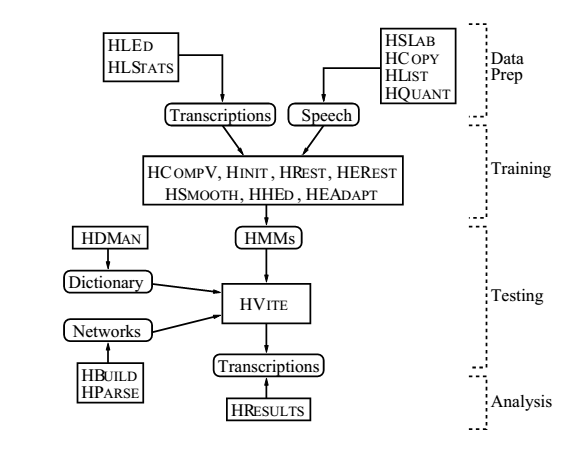
由HInit读入所有的引导训练数据，并迭代计算一组初始参数值。通过HInit计算的初始参数值由HRest进一步重估，直至收敛。

1. 测试

HVite使用令牌传递算法来执行基于维特比的语音识别。HVite把输入看作是一个网络，该网络描述了所容许的单词序列，并定义了每个单词如何发声的字典和一组HMM。它将单词网络转换成音素网络，之后在每个音素实例附上合适的HMM定义，以实现运转。接着就可在一系列存储的语音文件上或者直接的音频输入上进行识别。

1. 分析

基于HMM的识别系统的性能评估由HResults完成。

图7

经过需求分析后，本课题中对HTK工具箱的使用主要如图8所示，对一些手机端上不常用的功能进行了裁剪，使得本系统更为小巧稳定。



图8

## 4 进度安排

|  |  |
| --- | --- |
| 2014.11-2015.1 | 查阅文献，收集资料，熟悉课题，完成文献翻译和开题报告 |
| 2015.2-2015.3 | 系统需求分析，系统概要设计 |
| 2015.3-2015.4 | 系统详细设计 |
| 2015.4-2015.5 | 系统实现和测试 |
| 2015.6 | 撰写论文，参加论文答辩 |

## 5 参考文献

[1] 李兴华，王月清．Java Web开发实战经典基础篇[M]．北京：清华大学出版社，2010．90~324．

[2] 赵俊峰，姜宁，焦学理，