基于手写笔迹认证的Linux登录系统

专业：计算机科学与技术

学生：靳博 指导教师：于中华

**摘要** Linux衍生系统主要使用口令密码来进行登录验证。口令密码有易窃取易伪造等缺点。而每个人的写字风格具有其生物特征，已经在很多情况下用来验证身份，拥有易用性高，不易在现代机器学习算法验证下成功伪造等特点。PAM（可插拔式认证模块）是用来认证的通用框架，在Unix-like系统中被广泛使用。本文中，我们通过成熟的机器学习方法，对移动设备获取的笔迹序列进行预处理、监督性学习，并对算法进行改进，使得对于手写笔迹的识别达到一定程度上满足应用标准的正确率。训练完成后，本文将这一技术移植到Linux系统的登陆认证上。通过编写可插拔式认证模块，使得用户可以自由注册笔迹、并使用笔迹验证来登陆所拥有的Linux系统。本文实现了手写笔迹输入，并能自由选择手写风格的安卓应用；实现了手写笔迹认证模型；实现了Linux认证系统模块以及Linux上的笔迹注册应用；实现了提供笔迹识别接口的服务器。最终完成了基于笔迹识别的Linux登录系统。

**主题词** 登录验证，PAM，手写笔迹，动态时间规整方法，随机森林

**目 录**

**一、 绪论** 4

1. 课题研究的背景和意义 4

2. 本文所做的主要工作 5

3. 本文主要结构 6

**二、 相关技术研究** 6

1. Linux认证方式概述 7

2. PAM认证机制研究 8

2.1PAM简介 8

2.2PAM接口结构 8

2.3PAM涉及到的文件目录 9

3. 手写笔迹识别认证 10

3.1认证训练过程 10

3.2机器学习方法介绍 12

**三、 认证登陆系统需求分析** 14

1. 总体描述 14

1.1开发目标 14

1.2用户特点 15

2. 功能需求 15

3. 外部接口需求 15

3.1输入输出要求 **错误! 未定义书签。**

3.2硬件与软件环境 15

3.3软件接口 16

3.4通信接口 16

4. 安全需求 16

5. 分析模型 16

**四、 认证登录系统系统设计** 18

1. 运行环境 18

1.1硬件环境 18

1.2软件环境 18

2.系统总体设计 19

2.1体系结构设计 19

2.2功能需求与系统模块的关系 19

3.功能模块设计 19

3.1主机验证模块 19

3.2笔迹输入模块 20

3.2服务器模块 20

**五、 主要功能实现** 21

1. 主机端应用实现 21

1.1PAM模块实现 21

1.2pam-aware应用的实现 23

2. 手写笔迹分类算法的实现 23

2.1 特征提取算法 23

2.2 线性分类器的选择与测试结果 23

3. 服务器端实现 23

3.1 数据库实现 23

3.2 服务器接口的实现 24

4.输入端安卓APP的实现 25

4.1手写笔迹的获取 25

4.2 Bézier曲线的实现 26

4.3遇到的问题和解决方案 27

**六、 总结与展望** 27

1. 项目进行中遇到的困难及总体设计思路 27

2. 本项目的主要创新性 28

3. 为解决问题和可优化部分 29

**参考文献** 31

**声明** 32

**致谢** 33

**文献翻译** 34

1. 绪论
2. 课题研究的背景和意义

过去的三十年，计算机高速发展，计算机系统的复杂性也与日俱增。随着计算机在人们生活中扮演越来越重要的角色，它的不可或缺与无微不至，愈发凸显具有可操作性强、安全度高，方便简洁的安全保障措施的重要性。

传统的身份认证方法主要有三类，基于知识的方法，比如最常见的密码等；另外基于信物的方法，比如身份证、以及前段时间聚集了公众视线的NFC卡解锁等；再有即为近来越来越广泛采用的基于生物特征的方法。阿里巴巴公司在2015年推出人脸识别进行支付验证，将基于生物特征识别的验证方法的安全性与应用场景的范围都推向了一个更高的台阶；即使存在以照片甚至3D照片浑水摸鱼的风险，在新型技术上的前进，让我们的生活充满现代感的同时，也享受了科技带来的便捷。另一项广为应用的生物特征极为指纹特征。现在广大公司、科研院所都大幅配备了“指纹签到”机。其方便简易的认证方式，收到广大师生的喜爱。进入大众视野的还有依托腾讯公司得到广泛使用的“声音解锁”功能，在微信等应用中以双因子加锁的方式保障应用安全。其他被广泛应用的生物特征还有视网膜认证等。由于其对验证设备条件的要求，没有被广泛部署到我们的日常生活中。

生物特征识别无疑已成为身份认证研究的一大分支，由于其具有的口令、身份证等验证方式不具有的注册简单、使用方便的优势外，还有不易窃取，不易模仿等等特点。生物特征识别利用人与生俱来的生理特征，或者在平时的长期生活中，逐渐培养起来的一种生活习惯。诸如笔迹、步态等。其中手写笔迹识别无疑是诸多生物特征中比较具有可行性的一种。

手写笔迹的甄别训练，已经有四十余年的历史，在线手写笔迹的认证方式，随着计算机硬件日新月异的发展，其准确率和应用范围也越来越高。在手写设备无限普及的今天，智能手机在中国达到了96部/百人的高占比，进行手写笔迹的认证方式在硬件费用上越来越有可行性。而另一方面，除了硬件方面的低费用特点外，在线手写笔迹进行身份认证的另一大优势也不容忽视，就是手写笔迹，即一般情况下的笔迹，已经在我们平时生活中进行了普遍地应用，几乎无处不在的签名场景，使得我们的手写笔迹虽然经常变化风格，但在某一些文字上，能够保持相似的风格习惯；而且长期的在纸质材料上进行签字认证，也使得人们对于签名认证这一方式并不陌生，甚至有天然的顺理成章感。

手写笔迹不止局限于签名，很多人由于姓氏复杂，往往会选择以某些特定字符作为身份认证口令。所以本文题目为手写笔迹的认证，比起签名认证的说法，更加合理切题。

手写笔迹在认证安全领域扮演越来越重要的角色的同时，Unix-like系统的广泛使用也随着计算机的普及达到了很高的程度。目前Linux-like系统的使用率已经位居所有操作系统中第一位，仅android系统一项即达到54.16%的占比。基于Linux的众多发行版中，Ubuntu以对普通用户的友好界面，已经成为后起之秀，以每年2000,000实例的速度增长。其具有的广泛用户基础，现有的贫瘠认证方式，以及其开源特性，都为在Linux系统上开发基于手写笔迹的登录系统形成了非常好的先决条件。在Linux系统上开发一套可信度高的身份认证方法，既是一项有趣的科研探索，也无疑是一次具有实用价值的产品开发。

1. 本文所做的主要工作

Linux上的可插拔式认证模块（Pluggable Authentication Modules，以下简称PAM）是Unix-like系统上被大量采用的认证框架。目前主要的Linux发行版如Ubuntu，Debian，Mint，CentOS等系统五一例外使用PAM进行身份认证。本文基于对PAM的研究，和笔迹识别系统的研究现状，主要在以下几个方面进行了原理研究和相关应用的编写。

1.对PAM的详细研究和PAM模块的编写

PAM是Linux系统集成的系统模块之一，在PAM项目较为活跃的年代，留下了邮件列表和相关文档。但这些文档有两个主要问题，1.由于PAM开源项目的性质，文档大多语焉不详，大而化之，对于细节没有详细的阐述，只有思想性或设计性的概述。2.由于PAM项目现在已不再活跃，在互联网共享知识的高发期，网络上并没有除了官方文档外的PAM资料，同时的现有官方文档虽然没有错误，但对于很多问题没有更进一步的解释。

本文作者通读了近200页英文文档，与Ubuntu、CentOS等主要开源论坛资料，透彻详解了Linux-PAM的工作流程，对PAM认证过程进行了深入研究与思考。编写了PAM系统模块，并对安全性移植性等进行了考虑。

2.对PAM-aware应用的编写

在进行手写笔迹的认证登录中，注册模块是不可或缺的重要一环。由于本文基于Linux系统的特殊性，注册应用要充分考虑Linux系统多用户特征，在进行注册的同时，使得用户受到Linux系统的管理。

本文实现了PAM-aware的注册应用程序，利用系统开发库，使得注册用户与Linux用户表不产生冲突，并被Linux用户管理进行统一分配。

3.对手写笔迹认证的研究与训练

手写笔迹、在线签名的研究有几十年的历史。本文在尽量结合当前较新成果的前提下，使用随机森林、主成分分析等方法，对手写笔迹进行了训练。毫无疑问，手写笔迹认证是机器学习中的分类问题，本文实现了对手写笔迹在移动设备上的提取、预处理、模型训练等步骤，并在预处理、特征提取、模型训练等步骤中有自己的创新。

4.手写笔迹的输入端即安卓应用的编写

基于第一节所叙现状，本文利用安卓系统API，获取在安卓系统的触摸屏上进行操作的X,Y坐标以及压力值。完成了服务模块，注册模块和认证模块的编写。

值得一提的是，为了解决在安卓设备上广为人知的手写延迟问题，本文使用了局部矩形刷新法，达到了安卓设备手写输入的流畅使用。

5.手写笔迹的多风格输入功能

值得一提的是，本文利用Bézier曲线算法，实现了类毛笔的写作风格，符合中国使用者的书写习惯，在使用中进一步彰显确定其写作风格，达到更好的实用性。

本功能利用Bézier算法实现了五种不同毛笔风格，分别是“傅青主草书笔锋”，“王羲之行书笔锋”，“赵孟兆楷书笔锋”，“钟繇小楷笔锋”，“颜真卿大楷笔锋”。

6.实现了提供手写笔迹认证的服务器

在第三项训练结果的基础上，本文将训练模型序列化，实现了服务器接口，提供注册和验证两个主要功能。

1. 本文主要结构

本文共分五部分。

第一部分为绪论，简述了本课题研究的背景以及本文主要工作。

第二部分为本课题相关主要技术研究。概述了三个主要技术的应用背景，并对其应用情形与使用方法进行了详细研究。

第三至第四部分为本课题的需求分析与系统设计。本文使用用例图、顺序图等，记录了本项目编写过程中的需求和设计步骤。

第五部分是最终实现结果的呈现。经过近四个月的编写，本文实现了需求与设计模块的所有功能，并且在手写笔迹的机器学习算法方面有自己的创新。本部分详细说明实现中遇到的困难以及最终的实现情况。

最后部分即第六部分为对整体项目的总结，对实验过程中遇到的问题、困难、未解决问题均进行了有效反思，同时面对今后的完善提出了整体展望。

1. 相关技术研究

1. Linux认证方式概述

身份管理，也成为身份和访问管理，在计算机安全方面，被定义为“使正确的人在正确的时间和正确的理由获得正确的资源”的安全和商业几率。他解决了越来越多异构技术环境中对安全的保障，并满足越来越严格的符合规定的要求。一般包括四部分功能，即：

纯身份功能：创建，管理和删除身份，而不考虑访问权限或权利。

用户访问（登录）功能：例如：智能卡及其与客户登录到服务或服务（传统视图）使用的关联数据。

服务功能：为用户及其设备提供个性化，基于角色，在线，按需，多媒体（内容），基于在线的服务的系统。

身份联盟：依靠联合身份验证用户而不知道他或她的密码的系统。

对于操作系统安全的保证，身份管理中的身份认证身份认证是重要机制之一。在恶意用户、恶意攻击中，大部分系统通过设定的认证机制来进行用户身份的核查；通过用户身份认证，来作为用户进入系统的判定条件。所以身份认证是系统安全第一道防护措施，也是最重要的防护措施之一。

在系统安全保障中，关于身份认证已经产生了多种多样的机制，不管是认证理论还是认证技术都得到了突飞猛进的迅猛发展。比如口令机制，应用对称加密或非对称加密算法的各个机制，Kerberos认证体系等等。

需要进行身份认证的入口登录服务有很多，比较常见的有login，ssh，telnet等。在最初的Unix系统中，当认证算法进行了发展，或就有算法被破解时，经常需要从新改写整个应用程序来部署较新的认证机制。这种方法认证机制有两个主要缺点：1、与相关应用耦合性高，低内聚，代码复用性差，几乎没有达到模块化。2、需要为每个应用单独维护数据库，数据库开销大维护复杂。1995年，由Sung公司的Vipin Samar和Charlie Lai提出了可插拔式认证模块（PAM）。并将第一版PAM成功部署在Solaris系统上。PAM是将多个低级认证方案集成到高级应用接口（API）中的机制。它以来认证的程序独立于底层认证方案进行写入。于1995年10月在RFC86.0中提出，在那次会议中被采纳为通用桌面环境的认证框架。

作为独立的开原基础架构，PAM在1996年8月首次出现在Red Hat Linux3.0.4中。目前应用PAM认证机制的有DragonFly BSD，FreeBSD，HP-UX，Linux，Mac OS X，NetBSD和Solaris，有符合GPL协议的OpenPAM。

PAM已经是Linux Kernel发行版不可分割的一部分。在结构上位于Technical-Components-Security Modules条目下。

2. PAM认证机制研究

2.1PAM简介

Linux-PAM是一系列动态函数库组建，使得本地系统管理员可以选择应用如何认证用户。

换句话说，在不重新编译一个使用PAM的应用程序的情况下，是完全可以在不同认证机制中进行选择的。我们可以更新整个认证系统，却不改变应用本身。

上文已提到，应用如果要求对用户进行认证的话，需要使用特定的认证机制进行编译。比如，在传统UNIX系统中，用户通过输入正确密码进行身份确定。密码会被两位加盐前缀加密（见crypt(3)）。如果该加密后的密码，与系统密码数据库中用户条目第二部分相同的话，那么用户将被认证。在这类系统中，大多数（不是所有形式的权限）权限都是基于这种单一的认证方案授予的。权限以个人用户标识符（UID）的形式和各种组的成员身份来表现。基于用户的个人和组，服务和应用程序可以进行标识。传统上，根据group文件中的条目分配组成员的资格。

PAM项目的目的是将权限授予软件的开发与安全、适当的认证方案开发分开。通过提供应用程序可能用于请求用户进行身份验证的函数库来实现这一方案。在本地配置文件pam.conf中，选择本地可用的认证模块来验证用户请求。

Linux-PAM模块可以被Linux-PAM（以下简称PAM）的接口调用，并加载运行。PAM函数库可以在配置文件目录下，对配置文件进行修改，从而通过使用本地认证模块，来对用户的认证请求进行处理。模块均为链接库形式。一般为动态链接库，当动态链接方法不能使用时，可以使用dlopen函数，来进行自动地静态链接。

（这里是图片）

如上图所示，对于一个应用来说，如果要进行认证过程，该应用需要先调用PAM接口，然后通过PAM接口，读取配置文件，来进行PAM函数库的定位、加载、以及对PAM模块的调用。需要注意的是，在模块的编写过程中，模块的功能需要尽量保证，除去一些必须的操作，直接的目的，如键入密码之类，不与所服务的应用进行直接的调用，而应利用回调函数pam\_conv()来进行。

2.2PAM接口结构

PAM模块由四个独立部分组成即：认证（authentication）账号（account）会话（session）密码（password）。四个服务模块互相独立。在编写某一服务模块的时候，不能将此模块的运行依托于其他模块的运行成功。

如下图所示，四个模块主要负责功能大致如图。在本文中，主要用到PAM中的认证管理（红色方框内）部分。

（这里是图片）

账户管理（account） 该模块类型执行基于非验证式的帐户管理。其根据一天中的时间以及当前可用的系统资源（最大用户数）、申请人用户的位置（root仅允许在控制台登陆）等来限制/允许访问服务。

认证管理（auth） 该模块类型提供了认证用户的两个方面。首先，它通过指示应用程序提示用户输入密码或其他识别方式，来确定用户是否为其所声称。其二，该模块可以通过它的凭证授予属性来授予组成员身份或其他权限。

密码管理（password） 该模块类型是更新与用户相关联的认证令牌所必需的。通常，每个基于“质询/响应”的认证类型（auth）都要用到该类型的一个模块。

会话管理（session ） 这种模块类型与在提供服务之前或之后需为用户完成的事情相关联。这样的事情包括记录关于用户打开或关闭某些数据交换的信息，安装目录等等。

2.3PAM涉及到的文件目录

PAM是Linux内核的主要安全构件之一，PAM的安装和使用涉及到许多目录结构。现总结如下：

表1 PAM所涉及文件目录

|  |  |  |  |
| --- | --- | --- | --- |
| 文件功能 | 文件目录 | 备用目录 | 文件名 |
| PAM配置文件 | /etc/pam.conf | /etc/pam.d/ | login sshd sudo … |
| PAM模块设置文件 | /etc/ | /etc/security/ | login.defs nologin limits porttime issue securetty enviroment motd… |
| PAM可执行文件 | /lib/secerity/ | /lib/x86\_64-linux-gnu/security/ | pam\_util.so pam\_mail.so … |
| PAM读取文件 | /etc/ |  | passwd shadow … |

1. 手写笔迹识别认证

3.1手写笔迹认证现状

手写笔迹一般有两种认证方法。分别是基于参数的方法和基于函数的方法。或者说局部方法或全局方法。前者根据在输入信号中可以提取到的属性来作为新的特征，进行训练、分类等。常见的方法有，使用自编码器从手写笔迹信息学习特征、使用离散余弦变换、小波变换、傅里叶变换等技术生成特征。其优点为计算量小，很难根据数据集还原出原本的信号信息等优点。

基于函数的方法则使用整个序列来作为待训练的特征，而不是某些计算得出的属性。比如将其作为SVM的核函数来计算最长公共子序列，以及如本文采用的使用动态时间规整来解决。

但无论哪一种方法，都需要进行本部分第一节认证训练过程。只在步骤的内容选择上有差异之分。

当然，手写笔迹认证在现阶段发展还因为很多因素致使其训练结果及广泛应用被限制。其中主要原因有，1.没有较完善的可供训练数据集。2.手写笔迹本身容易受个人心情、状态的影响。

（这里是图片）

关于第一个问题，我们使用SUSIG数据集，尚未解决中文数据集较少问题。关于第二个问题，我们应用了三个解决方案。1.友好可自由转换笔锋的输入模型，强化书写风格。2.可以重复注册，来确保时效性。3.可实行双重认证机制，在手写笔迹无法识别时，应用传统方式解锁。

3.2认证训练过程

如图所示，一般说来，经由输入设备输入的手写笔迹，要经过以下几个阶段处理，然后使用不同的分类算法训练分类器，根据一定的评判标准，得到最终的结果。

（这里是图片）

3.1.1数据获取

数据来源，即手写笔迹序列的来源一般是通过特定硬件输入。硬件分为三类，专门的手写输入板，以及普通的带有触摸屏的移动设备。需要注意的是还有用电容笔进行操作与否的问题。其中比较特别的还有使用鼠标作为输入源（论文[7]）。

毫无疑问，手写输入板要比普通设备有更高的采样密度，更高的准确性。但随着硬件的发展，现在一般智能手机也已经满足采样要求，甚至相比早期的专业输入板也不遑多让。

本文中的训练数据来源是SUSIG数据库。SUSIG（Sabanci University Signature）是一款在线手写笔迹数据库。与MCYT数据库和SVC数据库相比，其主要优点在于，开源、免费，数据真实，获取数据方式为多任务获取，更加贴近真实情况。该数据库的数据由两部分组成，即用普通墨水屏收集的数据以及用专业压力敏感屏收集的数据。每个数据组由20个真实笔迹和10个伪造笔迹组成。

每个笔迹由5个属性组成，分别是x和y坐标，时间戳，压力水平，和一个向上或向下的指示器。

在本文实现系统中，数据由普通智能手机屏幕收集。每个笔迹由四个属性组成，分别是x和y坐标，时间戳和压力水平。

3.1.2预处理

由于手写输入设备硬件原因，捕捉的输入信号经常会有不完整情况或噪音污染。另外，输入笔迹由于输入设备或用户当时心情影响，很容易导致输入笔迹大小不一，真姓名间距离过大等问题。故此，对输入信号进行预处理已经成为机器学习领域约定俗成的第一步。

常见的预处理方法比较，以及本文是否使用该方法现列表如下。

表2 笔迹识别预处理方法比较以及本文是否使用

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 内容 | 效果 | 缺陷 |  |
| 删除冗余点 | 使用高斯滤波或只是简单删除同一、相近时间戳的输入点。 | 可以加快处理速度。 | 会一定程度上丧失数据特征。 | √ |
| 大小归一化 | 将输入信号强制归一化到某个大小区域内。 | 可以加快处理速度，保留真伪笔迹之间区别的同时，能够减小真笔迹之间距离。 | 丧失大小特征。 | √ |
| 位置归一化 | 将输入信号强制归一化到某个中心位置。 | 可以加快处理速度，保留真伪笔迹之间区别的同时，能够减小真笔迹之间距离。 | 与大小归一化步骤重复，可合并处理。 | X |
| 时间归一化 | 将时间戳归一化到某个区域内。 | 可以加快处理速度。 | 丧失了大部分特征。 | X |

3.1.3特征提取和特征选择

特征向量应尽可能在保证训练、预测时间的基础上，包含输入数据，X，Y，T，P显示的所有特征。

为了包含时间特征，我们引入VX和VY。

在使用这些特征的时候，目的是为了判断某一待验证数据，距离真伪数据的距离，亦即一个分类问题。那么如何计算距离，则是应该考虑的首要问题。用什么来代表与数据集的距离，从而输入分类器进行训练，是第二个问题。

在这里，我们使用比较常用且成熟的动态时间规划（DTW）算法。在很多已发表论文中使用该算法进行训练。该算法广泛应用于离散曲线间相似度近似问题。拥有准确率高，很少过拟合，自动匹配特征，忽视微小差距，计算速度快等多个特点。

第二，在计算完成两者间距离后，我们需要得出一套机制，来确定如何利用数据集间的距离。其中比较常用的方法有，选择最有代表性的一个笔迹，或者一个笔迹集，以与模板笔迹或笔迹集的距离计算，乘以一定权重，作为该数据的一个特征。

在选择笔迹集时，有直接认定法和重复选择模板法。根据论文，我们选择直接认定法。即直接认定设定数目个笔迹作为笔迹集。

有些论文还使用最有代表性笔迹，来保证最终的验证中，可以避免直接指定笔迹集带来的问题。

选择最有代表性笔迹时，一般将各信号简单相加，来选择与所有训练笔迹想加距离最小的手写笔迹，作为模板笔迹。也可以将信号乘以其权重进行选择。

第三，在计算完成特征向量时，该如何确定各个权重，是最后需要考虑的问题。关于特征向量的权重分为两个部分，亦即两个不同级别的部分。其一，如何确定VX，VY，X，Y，P这五个元素在代表数据时的权重。其二，如何确定某一元素中，计算出的各个距离的权重。

关于第一部分，我们使用特征向量进行训练，则问题转化为，如何去除冗余特征，从而简化计算。

关于第二部分，我们有如下几个选择，分别是平均值，中位数，最大值，最小值。以待计算数据与模板数据、引用数据集的DTW距离的平均数、中位数以及最大值最小值，来作为特征向量的待选呈现。

本文使用主成分分析算法进行特征向量选择。也可以人工选择，用实验结果判断优劣。由于本文设计的变量非线性相关，不能使用Pearson等方法进行选择。

需要注意的是，在计算特征向量时，由于各个向量大小、数量级不同，呈现结果也有区别。这种区别十分不利于之后的分类训练。有区分度的特征可能由于数量级太小而被忽视。所以我们这里使用基向量，进行归一化处理。基向量为所有引用数据集产生的特征向量。

由于基向量来源于真实笔迹。所以在生成的特征向量中，源信号越接近真实笔迹，则其越接近基向量，归一化后越接近1。反之亦然。

3.1.4选择分类器

在笔迹识别领域，有很多常用分类器，比如隐马尔科夫链，支持向量机，贝叶斯分类器等。其中比较好的效果使用为PCA算法进行处理后，使用普通线性分类器；还有使用随机森林算法，在一些论文中得到很好的评价值。

在本文中，我们简单对比了一些分离器，包括深度为五层的深度学习算法，随机森林分类器等。最终选择效果最好的为先使用主成分分析进行特征选择，再将结果输出到随机森林进行训练这一组合。

在本节的第二部分，将有机器学习方法介绍。

3.1.5评价标准

目前的笔迹识别认证中，较为常用的评价标准有两个，即真实笔迹的错误拒绝率( FRR )和伪造笔迹的错误接受率( FAR )。

由于FRR和FAR具有负相关性，所以在一些论文中，也会使用FRR与FAR相等时的数据，等误率（EER）来作为评判标准。

FRR=

FAR=

3.3机器学习方法介绍

机器学习致力于研究通过计算的手段，利用过去的经验，即数据，来改善系统自身的性能。机器学习是人工智能的一个分支，一般可以分三两类，监督学习无监督学习与半监督学习。无监督学习与半监督学习都针对部分或全部数据集没有分类标签的情况。监督学习主要指从所提供的训练数据集中学习，从而得到一个函数。该函数对新数据有预测功能。其训练数据应包括两部分，特征与分类标签。

由于本文需要对笔迹进行真实、伪造分类，故所用到的机器学习模型主要是监督学习。常用的监督学习为回归（regression）和分类（classification）。同时，在对特征进行提取时，还用到了无监督学习模型。

本文主要利用Scikit Learn（下简称sklearn）库中集成的机器学习方法。sklearn是Python上成熟的机器学习库，以功能强大，实现优质，代码简洁，与时俱进著称。

3.2.1主成分分析

在进行分类模型训练之前，特征提取是决定分类速度与分类结果的主要因素。除深度学习外，大部分机器学习模型不具备特征提取的功能。这使得需要我们在正式的机器学习训练之前，先进性特征选择步骤。

在选择特征时，一般的方法有，基于经验法与基于理论法。基于经验进行选择，又称为专家选择。在某个领域比较熟悉的专家在选择特征时往往得心应手，可以很轻易的去除不影响分类结果或者会混淆分类结果的特征。基于理论的特征选择中，使用主成分分析方法是其中比较常用的一种。

当面对大量特征，不知如何选择时，往往可以使用主成分分析（PCA）进行降维。在向量空间，必定存在一个超平面，当将向量在该空间进行投射时，可以使得数据点可以被最优的分开。

主成分分析被广泛地用于压缩、可视化以及机器学习中。一般在将高维信息压缩至低维时，不但不会丢失很多信息，还可以使训练速度加快。并且由于减少特征的数量，在减弱过拟合中也有很好的效果。

其算法描述如下：

|  |  |
| --- | --- |
| 对特征集F进行主成分分析 | |
| 1：  2：  3：  4：  5：  6：  7： | Input：特征集F={f1 f2…fm} 降维后的维度k  Output：降维后的特征集D={d1 d2…dn} |
| function PCA(F，k)  for f in F:  f = f – 1/m(fi)  lamda = Eigenvalue（F×FT）  D = Eigenmatrix(F×FT)  Sort(D,lamda)  retrun D(1:k) |

输入的降维维度k由用户事先决定，另一种方法是使用k紧邻分类器来进行交叉验证，从而得到较好的k值。

需要注意的是，在选择维度时，不要选择一维。因为特征向量一般会在一个子空间内旋转，单选择一个维度是毫无意义的。另外，在选择主成分分析方法时，要确定待处理数据确实需要维度化简，否则会达到事倍功半的反效果。

3.2.2随机森林

随机森林是上世纪八十年代产生的机器学习算法。与其他机器学习算法一样，运用控制方差作为目的，产生一系列决策树，进行投票决策。决策树是一种广泛使用的机器学习方法，是一种分类模型。

随机森林能够不但执行分类任务，还能进行回归。与此同时，随机森林还能作为几个模型之间的连接器，将模型组合成为一个更为高效的模型。在分类问题中，随机森林会输出系列决策树的多数决定。在回归中，输出则将会以平均值的形式输出。

随机森林模型对于我们来说是透明的，一般来说有三个选项可以被调整，用来控制其训练效果。1.最大特征数量。当数量增加时，可以提高性能，但牺牲了多样性。在减少数量时则有相反的效果。2.建立子树数量。数量增加时，可以改善结果，但会影响训练速度。3.最小样本叶子大小。较小的样本大小可能会较容易受噪声影响而过拟合。

决策树的优势在于构造简单、精度较高，其缺点一般认为难以明晓内部规律。随机森林的优势在于，有众多决策树进行投票，分类能力较高。从而可以看到，随机森林的性能受树的分类能力和树与树相关性关系很大。一般在选择可以处理连续属性和类别属性、需要解决分类不平衡等问题时有很大帮助。

在本文中，由于随机森林随机选择特征生成决策树，即使其具有选择特征能力，我们还是输入PCA处理后的特征向量，来加快速度的同时，增加其有效性。

3.2.3逻辑斯蒂回归

随机森林模型是一

3.2.4支持向量机

随机森林模型是一

1. 认证登陆系统需求分析
2. 总体描述

1.1开发目标

为用户登录解锁Linux发行版提供更为多样的解锁方式。用户可以使用手写笔迹进行解锁其拥有的Linux系统。

应用目标为拥有Linux系统并经常进行本地、远程登录的用户。

作用范围为拥有Linux系统并经常进行本地或远程登录的用户。

1.2用户特点

本产品主要应用于Linux发行版，使用者需要为Linux主要发行版客户，在进行计算机操作上不能存在障碍，使用者需要拥有可以安装配套APP的安卓设备。使用者需要能理解中文。

1. 功能需求

|  |  |
| --- | --- |
| 功能模块 | 功能描述 |
| 用户注册 | 用户运行Linux与安卓设备上相应注册程序进行注册，输入手写笔迹信息。 |
| 用户登录 | 用户使用与注册相同的手写笔迹进行Linux系统的登录。 |
| 用户删除 | 用户运行Linux用户删除程序，删除已注册用户，取消该用户使用笔迹解锁的权限。 |

1. 外部接口需求

3.1输入输出要求

1.输入：

（1）注册用户名必须为系统已注册用户名。

（2）输入笔迹不能过于简单。

2.输出：

无。

3.2硬件与软件环境

1.服务器

（1）处理器：Intel Core i5及以上

（2）内存容量：2G及以上

（3）操作系统：Windows XP及以上版本，Linux发行版等

2.主机

（1）处理器：Intel Core i5及以上

（2）内存容量：2G及以上

（3）操作系统：拥有PAM模块的Linux发行版

（4）网络：可进行联网通信

3.输入端

（1）屏幕：拥有可感知压力的触摸屏

（2）操作系统：Android4.1及以上发行版

（3）网络：可进行联网通讯

3.2软件接口

各模块过程之间采用函数调用、参数传递、返回值的方式进行消息传递。接口传递的信息将是以数据结构封装了的数据，以参数传递或返回值的形式在模块之间传递。

3.3通信接口

系统使用的网络协议为TCP协议，用于服务器、输入端和PC端之间的通信。

在输入端和PC端之间的通信，需要满足传输信息的可靠性和一致性，要满足通讯速度的要求并且支持多语言开发，并且封装了建立、处理、销毁链接的过程，使得通讯过程更简便。

在PC端和服务器端的通信中，需要可以满足分布式应答，便于硬件的拓展，并且满足传输速度与可靠性。

1. 安全需求

（1）主机端内存数据安全

主机端需要不记录任何与手写笔迹有关任何信息。在输入笔迹序列并处理后，应将所占内存进行数据填充。

（2）权限控制

用户将通过身份验证，只能对其拥有的身份进行注册。

（3）数据传输安全

应保证传输数据的主机端、输入端都是被认证的。所传输数据都是经过加密的。

（4）记录日志

本系统应该能记录主机端的所有操作，并能记录对服务器进行请求的所有可疑操作，包括本机错误、网络错误以及用户相关操作。并将这些记录保存在日志中。

5. 分析模型

1. 用户注册

（此处有图片）

用例：用户注册

主要参与者：主机系统使用者

前提条件：系统正常运行，使用者拥有主机系统账户和密码

迭代：需要使用系统

场景：

1.使用者打开主机，选择注册笔迹用户。

2.使用者输入该主机系统下已注册账户和密码进行身份认证。

3.主机系统提示验证通过。

4.使用者打开输入端，输入验证码、笔迹。

5.输入端显示已传输成功。

6.主机端显示注册成功。

（此处有用例图）

（此处有顺序图、流程图）

1. 用户登录

（此处有图片）

用例：用户登陆

主要参与者：主机系统使用者

前提条件：系统正常运行，使用者已注册笔记认证系统。

迭代：需要使用系统

场景：

1.使用者打开主机，进行登录。

2.使用者输入系统已注册用户名

3.主机系统提示用户名存在。

4.使用者打开输入端，输入验证码。

5.主机端显示验证成功。

7.使用者在输入端输入笔迹。

8.输入端显示传输成功。

9.主机端显示登录成功，并唤起下一个程序。

（此处有图片）

（此处有图片）

1. 认证登录系统系统设计
2. 运行环境

1.1硬件环境

1.1.1主机端

内存：12G

处理器：Intel Core i7-2600 CPU @ 3.40GHz × 8

系统架构：x86\_64

1.1.2服务器端

内存：6G

处理器：Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz × 8

系统架构：x86\_64

1.1.3输入端

硬件设备：nexus6-shamu

1.2软件环境

1.2.1主机端

GCC版本：4.8.2

内核版本：Linux version 3.19.0-25-generic

操作系统：Ubuntu 14.04 LTS

Python版本：2.7

Thrift版本：0.10.0

Protobuf版本：3.1.0

ZeroMQ版本：3.2.5

1.2.2服务器端

GCC版本：4.8.2

内核版本：Linux version 4.3.0-2-generic

操作系统：核高基桌面操作系统基础版 1.11.0

MongoDB版本：4.3.2

Python版本：2.7

Thrift版本：0.10.0

1.2.3输入端

内核版本：3.10.40-geec2459

安卓版本：5.1.1

Protobuf版本：3.1.0

ZeroMQ版本：3.2.5

2.系统总体设计

2.1体系结构设计

（此处有图片）

2.2功能需求与系统模块的关系

（此处有表格）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 主机验证注册模块 | 主机验证验证模块 | 笔迹输入注册模块 | 笔迹输入验证模块 | 服务器模块 | 网络模块 |
| 用户注册 | √ |  | √ |  | √ | √ |
| 用户验证 |  | √ |  | √ | √ | √ |

3.功能模块设计

3.1主机验证模块

1功能

提供主机上笔记解锁用户的注册以及已注册用户的登录功能。

2性能

要求可以进行与服务器和输入端的数据传输将数据保存在服务器数据库并为用户提供返回信息。

3输入项

用户名，用户密码，手写笔迹注册序列。

4输出项

为正则显示注册成功，否则显示注册失败。

5 相关技术

RPC传输使用ZeroMQ集合Google Protocol Buffer（以下简称Protobuf）技术，保证传输速度和可靠性。

6 接口

用户输入的用户名，笔迹序列，由Protobuf的结构体传入，由Thrift结构体传出。

7 限制条件

注册输入笔迹序列必须大于等于5。

8 尚未解决问题

并无尚未解决问题。

3.2笔迹输入模块

1功能

提供主机上笔记解锁用户的注册以及已注册用户的登录功能的笔迹序列的输入。

2性能

要求可以进行与主机端的数据传输。

3输入项

主机IP，端口号，验证码，笔迹序列。

4输出项

验证码，笔迹序列。

5 相关技术

RPC传输使用ZeroMQ集合Protobuf技术，保证传输速度和可靠性。

6 接口

用户输入的用户名，笔迹序列，由Protobuf的结构体传出。

7 限制条件

注册输入笔迹序列必须大于等于5。

8 尚未解决问题

并无尚未解决问题。

3.3服务器模块

1功能

提供用户名和注册笔迹的数据库存储，并提供笔记解锁用户的注册和验证接口。

2性能

要求可以提供可靠的连接服务，能够分布式访问。

3输入项

用户名，手写笔迹序列。

4输出项

输出服务状态，是否成功。

5 相关技术

RPC传输使用Thrift技术，保证传输速度和可靠性。

数据库存储使用MongoDB，便于分布式拓展。

使用sklearn机器学习算法。

6 接口

用户输入的用户名，笔迹序列以及处理结果，由Thrift结构体传出。

7 限制条件

数据库有足够空间。

8 尚未解决问题

并无尚未解决问题。

1. 主要功能实现
2. 主机端应用实现

1.1PAM模块实现

1.1.1通信接口实现

pam模块和移动端以及服务器端的通信，根据需求分析，需要分别满足不同功能需求的通信接口。pam模块和移动端的通信协议为Google Protocol Buffer自定义协议，提高压缩比；与服务器端的通信协议为Apache Thrift自定义协议，保证压缩比即传输速度的同时，简化服务器端代码逻辑，提高服务器处理性能和所提供服务的稳定性。

pam模块和移动端通信协议满足proto3接口定义语言。在两端传输中，要包含表明其身份的ID以及手写笔迹序列集。需要注意的是，笔迹功能即该笔迹为注册笔迹或认证笔迹由主机端直接决定，无需再次增加笔迹功能属性。

在手写笔迹序列集中，包含一个重复的数据结构手写笔迹序列。可以以任意数量重复，便于程序拓展。手写笔迹序列则包括重复的信号点，每一个信号点是输入设备采集到的一个数据点。每一个数据点的内容为数据点生成时间，该点输入设备自定义的相对X,Y坐标以及输入设备记录的压力值。

变量类型考虑目前安卓设备最大可以达到的精度。

pm\_m.proto相关定义内容如下：

package pm\_m;

message Point {

uint64 t = 1;

double x = 2;

double y = 3;

double p = 4;

}

message Signature {

repeated Point points = 1;

}

message Signatures{

string id = 1;

repeated Signature signatures = 2;

}

pam模块和服务器端的通信，满足thrift接口定义语言，由于主要需求相同,两通信接口除语法不同外，结构大致相同。

1.1.2编译语法

GCC是支持GPL3.0证书的开源编译器项目，拥有编译速度快、编译效果好，拥有自由的编译选项可以控制编译选项、优化效果、最终结果等优点。

其中用到的编译选项有：

-pthread 定义使用POSIX线程库所需的其他宏。

-shared 生成共享对象，然后可以与其他对象链接以形成可执行文件。并非所有系统都支持此选项。对于可预测的结果，您还必须在指定此链接器选项时指定用于编译（-fpic，-fPIC或模型子选项）。

-fPIC 如果目标机器支持，则发出与位置无关的代码，适合于动态链接，并避免对全局偏移表的大小有任何限制。

-dynamic 将标志-export-dynamic传递给ELF链接器，支持它的目标。这指示链接器将所有符号（不仅仅是使用的）添加到动态符号表中。此选项对于dlopen的某些用途或允许从程序中获取回溯是必需的。

pam模块的成功编译，需要以下几个步骤。

1.下载相关依赖库。

安装pam开发者依赖库

sudo apt-get install libpam0g-dev

安装Thrift，Protobuf以及ZeroMQ。

2.分别编译与移动端和服务器端通信协议满足pam模块运行环境主机架构的C语言版本。

thrift --gen cpp pm\_m.thrift

protoc --cpp\_out=. signatures.proto

3.编译pam模块。

g++ -fPIC -DPIC -shared -rdynamic -o pam\_kwe.so pam\_kwe.cpp signatures.pb.cc gen-cpp/\* -lzmq -lprotobuf -lthrift

4.将pam模块移动至系统pam模块指定目录下，并给予执行权限。

sudo mv pam\_kwe.so /lib/x86\_64-linux-gnu/security/

sudo chown root:root /lib/x86\_64-linux-gnu/security/pam\_kwe.so

sudo chmod 777 /lib/x86\_64-linux-gnu/security/pam\_kwe.so

5.修改login配置文件。

sudo patch -p0 login.patch

login.patch内容：

57d55

< auth sufficient pam\_kwe.so

1.1.3遇到的问题和解决方案

1.protobuf编写、编译后，C语言无法读取生成对象的成员函数。据分析由于数据结构定义问题，在最初的版本中，另外定义了结构体A，使得A包含重复的笔迹成员。再在Signatures中包含A和ID。这种定义不简洁，有冗余，而且由于这种冗余导致最终生成的成员函数无法正常读取。

2.在设置配置文件后，login显示打开session失败。在配置文件中，由于pam模块的调用方式是以模块栈的形式，从上到下依次调用。原则上所有模块互不依靠。但是在login中，我们的模块没有实现session功能，所以所处位置必须session之下；而要达到多重验证功能，又要位于所有auth配置之上。

3.ZeroMQ没有定时功能。本文使用多线程技术，完成定时功能。

1.2pam-aware应用的实现

1.2.1编译命令

g++ siguseradd.cpp signatures.pb.cc gen-cpp/\* -o siguseradd -lpam -lpam\_misc -lthrift -lzmq -lprotobuf

1.2.2遇到的问题和解决方案

1.显示链接错误。在编译pam-aware程序的时候，除了-lpam参数，还需要另外使用-lpam\_misc来支持conversation函数的传递参数数据结构的定义。

2.显示打开Socket错误。由于TCP内部协议，在绑定时不能使用127.0.0.1作为IP地址，需要改为\*来适配。

1. 手写笔迹分类算法的实现
   1. 数据获取与预处理

本文数据来源于第四章所述硬件设备。下图显示了最终进行验证用信号序列的可视化图片：

（此处是图片）

数据由4个属性组成，分别是时间戳，X、Y坐标以及压力值。

在本部分第二节，我们还提出了一种基于比划的计算方法。在该方法下，需要额外标记比划隔断，以进行后续的处理。

在选择预处理方法时，我们分别计算了，进行X，Y归一化，进行X，Y，T归一化以及不进行归一化下，笔迹序列特征的差距数据。

对于每一个信号S

S = (S-Smin)/(Smax-Smin)\*归一化标准（此处是公式）

为了减少运算量，在选择预处理方法时，我们不选择最终的评价标准来进行判断，而是使用特征选择方法。特征选择方法有多种，我们选择较为通用的卡方统计量进行计算。

输入的信号，会面临以下几种可能：是否进行预处理信号大小归一化，使用简单DTW，即一般DTW，使用本文提出的DTW，以及优化Threshold和Penalization参数之后的DTW算法。计算后的数据量，为了便于比较，均进行了数据归一化，使得矩阵模为统一为1。

表4 X信号在不同特征计算方法下归一化与否的卡方统计量

|  |  |  |
| --- | --- | --- |
|  | 进行大小归一化 | 不进行大小归一化 |
| 简单DTW | 0.8128106 | 1.02209779 |
| 本文提出的DTW | 0.60507707 | 0.67018404 |
| 参数优化后的DTW | 0.56222323 | 0.30796672 |

（见下节）由上表可知，参数优化后的DTW具有绝对优势。与第二章第三节的分析相同，不进行大小归一化，反而会拥有更好的结果。可以初步解释为，在SUSIG数据集中，收集到的笔迹为同一输入设备上，且收集时间相近，笔迹不会有明显的大小变化。不进行大小归一化明显会更大程度保存真假笔迹之间的不同。而在我们的日常应用中，必须要考虑输入设备不断更换的可能性，不进行大小归一化会影响到手写笔迹的处理。

* 1. 特征提取选择算法

手写笔迹识别中，特征提取算法的实现是比较重要的一步。

如第二章第三节所述。在手写笔迹识别中，如果选定了方法之后，还需要定一系列参数与方法细节。

在真实笔迹与伪造笔迹中（见下图），我们可以看到，伪造笔迹拥有更多的锯齿，并在时间长度上有很大区别。

（此处有图片）

很自然，我们选用优化的DTW算法。

该算法公式如下：

（这里是公式）

其中Dist()函数表示两个点之间的距离。

1.在选择DTW算法参数时，我们需要分别确定特征向量的Threshold以及Penalization。在选择时，由于待确定参数过多且硬件瓶颈运行过程较长，我们只使用一组笔迹进行训练，来选择Threshold和Penalization。凭训练结果即对照评判标准好坏取的尽量最优参数。

2.在提取待处理变量时，通过理论分析，我们最终确定X，Y，P，VX，VY几个待处理变量。

需要注意的是，再设计计算算法时，要考虑不同长度的签名，会拥有大小不一的特征值，在这里，有必要进行归一化，使得每组签名具有相同量纲的特征值。虽然特征向量中的特征不一定具有相同权重，但计算出的由于其量纲不同导致的权重不同必定不是我们需要的结果。

计算特征向量算法描述如下：

SgnX = [temSgn，AvgSng，MaxSng，MinSng，MedSng]

SngX[i] = DTW(Sng)/baseVector

（此处是算法）

本文为提高运行速度与精度，在理论上分析，若计算复合速度：

V2 = VX2+VY2

可以减少一个运算变量的同时，复合两方向速度，体现X，Y的相关性。但使用复合速度V的缺点为损失方向信息。

在选择随机森林分类算法情况下1410个真实笔迹序列和940和超仿笔迹序列数据集，不同的速度计算方法实验结果如下：

表4 不同速度特征在相同分类算法下的训练结果

|  |  |  |  |
| --- | --- | --- | --- |
| 特征选择 | 运行时间 | FRR | FAR |
| VX，V，P | 8162.53秒 | 5.61% | 1.60% |
| VX，VY，P | 8069.88秒 | 5.68% | 1.49% |
| V，P | 5937.33秒 | 7.10% | 2.03% |

如图可知，虽然单独使用V运行时间大大减少，但相应的错误接受率和错误拒绝率也增加很多。将减少的运行时间平均到2350个数据，平均每个数据减少了1秒左右，而错误拒绝和错误接受数量则只分别增加大约20和15个。

故而在最终的版本中，我们选择综合速度V，来代替分别计算的VX，VY。

真笔迹差距越小，真假笔迹差距越大，并且运算时间越少，我们说表现出更好的效果。

3.由前文可知，待选择变量共有24个。在选择特征变量时，我们可以重复应用第一节的方法，也可以使用主成分分析方法进行自动选择。在此，我们使用主成分分析自动选择。

4.在选择模板笔迹时，最好的方法是，比较特征向量乘以特征的权重之后的值。选择模板笔迹算法如下：

WT = V\*WV

T = min(WT)

其中，特征的权重比较，我们可以依旧使用卡方统计量，按照一定比例相乘。但是卡方统计量的比例并不能代表特征间的比例，只能体现大小关系，而没有很强的线性关系。所以在这里，我们使用随机森林算法进行特征排序，更能体现特征间的比例特点。

在随机森林算法下，我们得到各特征重要值如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | X | Y | VX | VY | V | P |
| TMP | 0.045 | 0.03 | 0.03 | 0.005 | 0.07 | 0.06 |
| MIN | 0.055 | 0.08 | 0.05 | 0.01 | 0.06 | 0.025 |
| AVG | 0.06 | 0.04 | 0.06 | 0.02 | 0.04 | 0.04 |
| MED | 0.035 | 0.025 | 0.06 | 0 | 0.085 | 0.01 |
| SUM | 0.09 | 0.175 | 0.20 | 0.035 | 0.255 | 0.135 |

由上图可以发现，综合看来，如前分析的相同，速度占有更高的区分度。其中是VY比重最低，因为随机森林会降低具有相似作用的特征，比重低代表不重要，但其代表意义不绝对。

5.需要注意的是，还有一些变量没有进行考虑。比如，在选择模板集时，其判断标准的选择，甚至在开始提取特征的时候，使用全局特征还是局部特征，都没有一个很好的结论。目前的论文由于数据集的不同，其训练结果并不具有很好的参考性。

另外，还可以将签名笔迹按照比划进行分别运算。在一些文章中，按照比划进行标记，再运用隐马尔科夫链，获得了很好的效果。本文尝试了这种算法，但是依然使用DTW运算，并将结果简单相加。最终效果并没有不进行标记好。初步分析是因为拆分后会损失比划间的时间连接信息。但此法应该会由于对速度的精准表达最终提升性能。由于此方法需要进行参数调整研究，本文进行的简单实验没有参考性，故在此不列出实验数据，有兴趣者可以深入研究此法。

2.3 线性分类器的选择与测试结果

在选择线性分类器时，我们简单的试验了支持向量机方法，随机森林方法，逻辑斯蒂回归方法以及梯度下降方法。

对于随机森林等同时拥有分类和回归方法的分类器，我们使用回归方法，以计算其EER。

线性分类器的输入即为进行了上述处理的特征向量集以及标签集：

TrainX = [Vsig1,Vsig2…Vsign]

TrainY = [Lsig1,Lsig2…Lsign]

在验证时，其输入为经历了上述处理的特征向量：

VerifyX = Vsig

我们使用skearn机器学习库进行训练和认证过程，主要代码如下：

classifier = model.fit(TrainX,TrainY)

result = classifier.predict(VerifyX)

（修改格式，是否应用regression）

本文提出的方法，训练结果如下：

表5 不使用PCA进行特征选择

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类器选择 | 运行时间 | FRR | FAR | EER |
| 支持向量机 | 8162.53秒 | 5.61% | 1.60% |  |
| 随机森林 | 8069.88秒 | 5.68% | 1.49% |  |
| 逻辑斯蒂回归 | 5937.33秒 | 7.10% | 2.03% |  |
| 梯度下降 |  |  |  |  |

表6使用PCA进行特征选择

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类器选择 | 运行时间 | FRR | FAR | EER |
| 支持向量机 | 8162.53秒 | 5.61% | 1.60% |  |
| 随机森林 | 8069.88秒 | 5.68% | 1.49% |  |
| 逻辑斯蒂回归 | 5937.33秒 | 7.10% | 2.03% |  |
| 梯度下降 |  |  |  |  |

综合来看，本文最好的方法，比论文[7]下降了。

1. 服务器端实现
   1. 数据库实现

为了使得服务器端有更灵活的数据结构，更容易进行拓展，我们使用NoSQL数据库类型MongoDB进行存储。

由于MongoDB简单的语法，其支持不预先声明直接存储，同时支持多种数据结构，以文件问单位进行存储。故而本部分实现十分简单。

数据库Collection结构为：

{ \_id，name，handwrite }

本文使用MongoDB Python API pymongo进行数据库操作。

相关代码为：

def \_\_init\_\_(self, config):

self.config = config

client=MongoClient(\

self.config.DatabaseIP, \

self.config.DatabasePort)

db = client[self.config.DatabaseName]

self.table = db[self.config.DatabaseTableName]

def save(self, \_id, handwrite):

self.table.find\_one\_and\_update(\

{"name": \_id},

{"$set": {" handwrite ": handwrite }},

upsert=True)

def load(self, \_id):

cursor = self.table.find({"name": \_id})

if cursor.count() == 0:

return None

signatures = cursor[0][" handwrite "]

return handwrite

* 1. 服务器接口的实现

服务器端与主机端的传输使用Apache Thrift接口定义语言。Thrift将待传输 的结构化数据序列化并且提供RPC框架。相比于其通讯框架，具有速度快、内存占用低、数据传输简便、支持数据结构丰富、跨平台等等特点。

作为通讯框架，Thrift要预定义通讯协议，其主要定义内容如下：

namespace cpp pm\_m

struct Request {

1: string id,

2: list<Signature> signatures,

}

struct Ret {

1: bool success,

2: ErrorCode error,

}

service HandWriter {

i32 ping(1:i32 num),

Ret accountRegister(1:Request request),

Ret verify(1:Request request)

}

3.3遇到的问题和解决方案

1.在运行C语言版本时出现运行错误。在生成Thrift文件时，系统自动生成main函数，在运行时需手动删除。

2.在两端通信时，如果版本不一致，会返回处理错误并直接关闭连接。

3.在服务器与机器学习结构进行结合时，我们使用了cPickle的序列化方法，简化服务器端的工作量，从而不用经常进行分类器的训练。

4.输入端安卓APP的实现

4.1手写笔迹的获取

在安卓系统上，由于其与手写板的区别，我们一般需要该设备和操作系统版本提供触屏操作的相关信息接口，通过接口进行数据获取。

一般说来，安卓系统必须提供此类接口，因为系统开发者不能自己开发与硬件沟通的底层软件，而且作为应用程序，不能对系统源代码进行修改。安卓系统提供的触屏接口是MotionEvent类。所以我们可以直接利用此类，进行输入笔迹的X值，Y值，P值，并确定时间戳。

相关代码如下：

@override

public boolean onTouchEvent(MotionEvent e) {

float eX = e.getX();

float eY = e.getY();

switch (e.getAction()) {

case MotionEvent.ACTION\_DOWN:

getParent().requestDisallowInterceptTouchEvent(true);

tPoints.clear();

if (isDoubleClick()) break;

tLastTouchX = eX;

tLastTouchY = eY;

addPoint(getNewPoint(eX, eY));

if(mOnSignedListener!=null)

mOnSignedListener.onStartSigning();

case MotionEvent.ACTION\_MOVE:

resetDirtyRect(eX, eY);

addPoint(getNewPoint(eX, eY));

mMotionEventRecord.put(\

mMotionEventRecordIndex++,

new MotionEventRecorder(\

e.getEventTime(),

e.getX(),

e.getY(),

e.getPressure()));

break;

case MotionEvent.ACTION\_UP:

resetDirtyRect(eX, eY);

addPoint(getNewPoint(eX, eY));

getParent().requestDisallowInterceptTouchEvent(true);

setIsEmpty(false);

break;

default:

return false;

}

invalidate();

return true;

}

需要注意的是，由于Android系统为增加速度设计的机制，在本应显示的数据点外，还有一些（通常不超过2个）数据点，以history的形式，储存在MotionEvent变量的成员变量内。在服务器性能较快的情况下，可以捕捉并传输这些点，以获得更高的精度。

代码如下：

int hsize = event.getHistorySize();

for (int i = 0; i < hsize; i++) {

float hx = event.getHistoricalX(i);

float hy = event.getHistoricalY(i);

}

4.2 Bézier曲线的实现

在安卓手写输入板的实现中，由于Canvas在绘制时，会直接绘制MotionEvent捕捉到的数据点，绘制出的效果往往为简单的线条，不符合平时书写习惯。而笔迹解锁最重要的，即使使用户能尽量保持书写习惯。

所以在这里，我们使用升级的贝塞尔曲线方法，来完成视图的绘制。

一般的贝塞尔曲线，使用贝塞尔算法来平滑绘制曲线，减少锯齿现象。在这里，我们通过设定曲线宽度取值范围，并且计算捕捉线条当前速度，来决定曲线此时宽度，进行绘制。最终达到恰似毛笔效果的目的。

（此处是图片）

同时，我们提供不同的宽度范围和宽度增减速度参数，来达到产生不同毛笔效果的目的。

相关代码如下：

private void addBezier(Bezier curve, float swidth, float ewidth) {

float ow = tPaint.getStrokeWidth();

float wdelta = ewidth - swidth;

float steps = (float) Math.floor(curve.length());

for (int i = 0; i < steps; i++) {

float t = ((float) i)

float t = Math.pow(t,3);

float u = Math.pow(1 – i,3);

float x = u \* curve.startPoint.x;

x += 3 \* u/(1-i) \* i \* curve.control1.x;

x += 3 \* (1-i) \* t/i \* curve.control2.x;

x += t \* curve.endPoint.x;

float y = u \* curve.startPoint.y;

y += 3 \* u/(1-i) \* i \* curve.control1.y;

y += 3 \* u \* t/i \* curve.control2.y;

y += t \* curve.endPoint.y;

// Set the incremental stroke width and draw.

tPaint.setStrokeWidth(swidth + t \* wdelta);

expandDirtyRect(x, y);

}

tPaint.setStrokeWidth(ow);

}

private float strokeWidth(float velocity) {

return Math.max(tmaxw / (velocity + 1), tminw);

}

4.3遇到的问题和解决方案

1.在拷贝完毕传输手写笔迹信息时，传输结果为空。由于拷贝数据为浅拷贝，而每此手写笔迹拷贝后均会清零，导致多线程并行时，会发生传输结果为空的情况。解决方法为换为深拷贝。

2.在绘制视图时，有锯齿和迟钝发生，影响用户体验。解决办法为增加DirtyRect进行局部刷新，并且使用贝塞尔曲线，来达到平滑化目的。

3.在获得数据点时，采样量较稀疏。我们研究MotionEvent后发现，安卓将更多点隐藏在getHistory方法内。可以进行手动取出。

1. 总结与展望
2. 项目进行中遇到的困难及总体设计思路

在本项目从选题、确题、调研，到进行实验、进行测试，直至最后进行总结和完善，都存在遇到问题、再克服困难的过程。而在困难中的解决办法的设计则体现了本文的创新性。

从选题始，作者在具有实用性、创新性，能够应用机器学习方法，并较为深入操作系统这几项基本要求中进行权衡，努力兼容并举，进行选题确题的枚举与排除，最终确定了本课题。在网络上基本没有相似功能的项目实现，而在知网等学术数据库中，关于PAM的论述已寥寥无几，实现PAM模块的登录应用更是少见。针对Linux桌面安全，登录无疑是保障安全维护安全的第一道系统屏障。而笔迹识别是一项有着完备的研究，并存在可用于实践的很好结果的认证技术。本课题在创新性、实用性上都有较为突出的表现，对机器学习的应用使得本实践性课题还具有了一定的学术性。

在调研中遇到的主要问题是难以登堂入室、择门而入。在确立的选题后，Linux如何进行登录验证，即Linux登录验证的逻辑调研部分比较令人担忧。一方面，开源系统都存在很好的模块性，但同时也有很高的集成度。我们相信广受欢迎的开源系统Linux及其发行版本Ubuntu一定提供更改验证逻辑的接口。而另一方面，如果由于操作系统安全性保证的要求，对其登录验证的修改需要更改大量源码并重新编译某一重要模块，则本项目的移植性必回大受影响，从而影响其实用性。而对Linux系统，其wiki目录，开源论坛运行模式的种种不熟悉，则在一定程度上加重了调研的难度。其二，在笔迹识别算法的调研中，与之相对，则面对资料众多的问题。而机器学习毫无疑问是本科阶段没有深入接触的一个新领域。在对机器学习算法的调研中，也花费了大量时间精力。

本项目完成过程中遇到的主要困难在实验阶段。经历了需求分析后，系统如何设计，才能保证其安全性、实用性，让用户可以顺畅使用的同时，系统管理员可以更为简便的进行维护，是实验阶段主要考虑的问题。在保障实用性中，对笔记解锁模块发生错误时系统的可用性；模块之间的通信：通信速度和通信可靠性；系统防范基本攻击的能力，主要针对网络攻击，假冒用户攻击，Dos攻击等形式；考虑系统的可移植性与可拓展性，尽量集中系统功能，简化附加模块设计逻辑。考虑实际应用情况、机器学习模型学习能力、未来功能拓展等可能，对系统的总体设计进行不同程度的侧重。考虑用户的便携性（UI设计方面），不止于实验室情况，以发行版本要求进行系统设计与代码的编写，提供更为友好的用户界面，更为流畅的操作体验，更为优质的系统性能。

1. 本项目的主要创新性

在项目选题阶段，创造性的提出使用手写笔迹进行桌面登录解锁的想法。提出新颖的桌面解锁方式，基于PAM的同时，结合多项技术，最终完成此解锁系统。

在机器学习模块，本文在汲取前人研究成果的同时，首先提出了使用计算后的速度值来代替分别的VX，VY速度值，在一定基础上提升了验证速度，并保证等误率（EER）。本文首先提出了使用PCA进行特征选择，再用随机森林进行分类的方法，在一定程度上降低了等误率。本文还从理论和实践交叉分析了笔迹识别各步骤的处理方法优劣。另外，本文编写了图形化显示类，用来以多种方式将笔迹训练结果图形化展示。

在安卓输入应用中，本文首先实现了对各个不同书法风格的笔锋模仿，在业界广为使用的Bézier曲线算法基础上，结合作者书法功底，为中文解锁者更好的用户体验，结合人类在舒适环境内书写能更好保持个人行为风格的心理研究，进行了大胆创新。同时，作者优化了安卓设备对笔迹捕捉的性能，增加其采样点。

在笔迹识别系统的最终实现上，本文首先考虑了手写笔迹的安全性，考虑破解难度问题，在设备中初步集成了检验笔迹复杂性，即保证密码复杂性的功能。本文也考虑了系统的安全性与运算性能，使用Protobuf，ZeroMQ等工具组成传输模块。

1. 未解决问题和可优化部分

笔迹识别机器学习模型训练中，由于学术界现有的数据集较少，且主要为非中文书写数据集，导致训练完成后的分类器可能对中文手写笔迹不会有训练希望达到的最优的分类性能。

在今后的优化中，可以设计手写笔迹采集工具，搜集整理更多中文笔迹，进行模型的训练。

在移动输入设备与主机的通讯中，本文采取的是通过无线传输的TCP协议。为了进行移动输入设备和主机间有效通讯，有三种技术比较常见。

1.使用无线传输。

2.使用有线USB传输。

3.使用蓝牙传输。

三种技术各有优缺点。本文在使用无线传输的设计上，要求硬件设备支持WIFI功能，并且输入设备和主机必须处于同一局域网内，其三，对主机进行通讯必须手动输入主机IP与端口号。针对第三项要求，有可改进余地。

在手动输入IP与端口号时，虽然使用SharedPreferences技术简化用户使用步骤，仍然较显繁琐。以下提供一些优化思路的优缺点比较：

1.将设备注册与手写笔迹注册认证步骤分开。设备注册后进行注册或认证时，系统通过WIFI或蓝牙直接发送信息至移动设备，激活输入模块。

优势：私有输入设备，提高安全性。提高操作流畅性。

缺点；大规模应用时，对移动设备软性要求较高。

2.改变设备与主机单向通信的设计逻辑，统一由服务器进行信息沟通。

优势：无需繁琐配置步骤

缺点；设计不清导致后期维护移植等种种问题

3.主机生成含有IP和端口号的二维码，移动设备通过扫二维码代替手动输入。

优势：简化配置步骤

缺点；增加设备硬件要求，需要具备摄像头

参考文献

[1] Hintjens,P.(美)著,卢涛,李颖译.ZeroMQ:云时代极速消息通信库.第一版.北京:电子工业出版社，2015.3

[2]李航.统计学习方法.第一版.北京:清华大学出版社,2012.3

[3]聂晓旭,于凤芹,钦道理. 基于 Protobuf 的数据传输协议[J]. 计算机系统应用, 2015, 24(8): 112-116.

[6]Raymond,E.S.著,姜宏，何源，蔡晓骏译.第一版.北京:电子工业出版社,2012.8 [4]Alisher Kholmatov,Berrin Yanikoglu. SUSIG: an on-line signature database, associated protocols and benchmark results[J]. Pattern Analysis and Applications, 2009, 12(3): 227-236.

[5] Kenneth Geisshirt.Pluggable Authentication Modules the Definitive Guide to PAM for Linux SysAdmins and C Developers.First published. Packt Publishing Ltd.

[6] 吕伟航. 随机森林算法研究及改进[D]. 厦门大学, 2013.

[7]User Authentication Through Mouse Dynamics

[8] Alisher Kholmatov,Berrin Yanikoglu. SUSIG: an on-line signature database, associated protocols and benchmark results[J]. Pattern Analysis and Applications, 2009, 12(3): 227-236.

[7] Kholmatov A, Yanikoglu B. Identity authentication using improved online signature verification

method[J]. Pattern Recognition Letters, 2005, 26(15):2400-2408.

声明

声明

致谢

感谢于中华老师在我毕设选题阶段地悉心指导。不知多少次，不管是到望江校区基教楼的实验室，还是通过邮件、微信，于老师总会在忙碌的工作中抽出时间，对我的疑问耐心解答，对我的学习生活进行认真指导，使我受益匪浅。

感谢于老师实验室的众位老师师兄师姐，对我有问必答，友善相助。

感谢四川大学计算机学院的诸位老师，尤其是吴志红老师，张恩乐老师，潘薇老师对我生活学习的关怀；感谢段磊老师，章毅老师，吕建成老师，刘艳丽老师对我计算机学习的启迪；感谢冯子亮老师，胡术老师，王明辉老师对我无私的帮助。在四川大学四年的学习中，我要感谢熊勇老师、朱宏老师，卢晓春老师，罗以宁老师、胡大裟老师，蒋欣荣老师对我耐心地受业解惑。当然还要感谢为我授课的各位老师，积土成山积水成渊，正是四川大学计算机学院对我的培养，让我认识计算机、了解计算机、熟识计算机，最终可以独立完成本次毕业设计。

感谢中科院软件所总体部的贺也平老师，为我进行毕业设计指明方向。感谢周启明老师，在我毕设遇到问题时与我讨论，对我鼓励。感谢周老师组内的各位师兄，我平时生活遇到问题时总是能及时帮助。

感谢我的母亲，在我毕设阶段遇到困难，精神焦虑、紧张时，她安慰我给我鼓舞，让我能够专心于问题的解决，最终完成毕设项目的所有设想。

毕业设计的完成，离不开师长同学的帮助，在此对所有有意、无意对我提携帮助的人们再次表示感谢。

文献翻译

翻译