**2019年全国大学生信息安全竞赛**

**作品报告**

**作品名称： 一种基于机器学习的安卓多因素认证系统MVSA**

**电子邮箱： [a179746859@gmail.com](mailto:arya_lee@qq.com)**

**提交日期： 2019.05.29**

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。

2. 作品报告采用A4纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5倍行距。

3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。(本页不删除)

4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。

5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

**目 录**

[摘要 1](#_Toc15026276)

[第一章 作品概述 3](#_Toc15026277)

[1.1 选题背景 3](#_Toc15026278)

[1.2 研究现状 9](#_Toc15026279)

[1.3 特色描述 10](#_Toc15026280)

[1.3.1 多因素认证 10](#_Toc15026281)

[1.3.2 九宫格划屏密码输入键盘 10](#_Toc15026282)

[1.3.3 在安卓多因素认证系统这一案例中首次引入无监督异常检测模型 11](#_Toc15026283)

[1.3.4 用户操作无感 11](#_Toc15026284)

[1.3.5 用户使用便捷 11](#_Toc15026285)

[1.3.6 在多场景输入下系统的稳定性与可靠性 12](#_Toc15026286)

[1.3.7 低资源开销 12](#_Toc15026287)

[1.4 市场分析 12](#_Toc15026288)

[1.5 本章小结 14](#_Toc15026289)

[第二章 作品设计与实现 15](#_Toc15026290)

[2.1 系统功能 15](#_Toc15026291)

[2.2 系统依托的安卓传感器介绍 16](#_Toc15026292)

[（1）安卓基础传感器 16](#_Toc15026293)

[（2）安卓触屏接口 17](#_Toc15026294)

[2.3 身份认证算法和工具 18](#_Toc15026295)

[（1）支持向量机 20](#_Toc15026296)

[（2）无监督异常侦测之——自动编码器 21](#_Toc15026297)

[2.4 MVSA系统的运行流程 22](#_Toc15026298)

[2.5 MVSA系统模块介绍 23](#_Toc15026299)

[2.5.1用户数据采集模块 23](#_Toc15026300)

[2.5.2用户数据训练模块 32](#_Toc15026301)

[2.5.3用户验证登录模块 37](#_Toc15026302)

[2.6 本章小结 39](#_Toc15026303)

[第三章 作品测试与分析 40](#_Toc15026304)

[3.1 测试数据集 40](#_Toc15026305)

[3.2 测试指标 40](#_Toc15026306)

[3.3 测试方案 42](#_Toc15026307)

[3.3.1 二分类模型的测试 42](#_Toc15026308)

[3.3.2 无监督异常侦测的测试 43](#_Toc15026309)

[3.4 测试环境 43](#_Toc15026310)

[3.4.1硬件配置 43](#_Toc15026311)

[3.4.2软件配置 43](#_Toc15026312)

[3.5 测试过程及结果 44](#_Toc15026313)

[3.5.1 按键密码测试 44](#_Toc15026314)

[3.5.2 九宫格密码测试 49](#_Toc15026315)

[3.6 本章小结 51](#_Toc15026316)

[第四章 创新性说明 52](#_Toc15026317)

[4.1 创新性 52](#_Toc15026318)

[4.1.1 多因素认证 52](#_Toc15026319)

[4.1.2 九宫格划屏密码输入键盘 52](#_Toc15026320)

[4.1.3 在安卓多因素认证系统这一案例中首次引入无监督异常检测模型 53](#_Toc15026321)

[4.2 实用性 53](#_Toc15026322)

[4.2.1用户操作无感 53](#_Toc15026323)

[4.2.2 用户使用便捷 53](#_Toc15026324)

[4.2.3 多场景下稳定可靠 54](#_Toc15026325)

[4.2.4 低资源开销 54](#_Toc15026326)

[4.3 本章小结 54](#_Toc15026327)

[第五章 总结 55](#_Toc15026328)

[5.1 工作总结 55](#_Toc15026329)

[5.2 下一步工作计划 56](#_Toc15026330)

[5.3 心得体会 56](#_Toc15026331)

[参考文献 57](#_Toc15026332)

图目录

[图 1 智能手机使用需求 3](#_Toc14967110)

[图 2 中国电子支付的快速增长 4](#_Toc14967111)

[图 3 网民多集中在 2001-5000 元月收入区间 5](#_Toc14967112)

[图 4 非高知类网民占据巨大多数比例 5](#_Toc14967113)

[图 5 中国网民安全意识薄弱 6](#_Toc14967114)

[图 6 支付宝的部分安全设置 6](#_Toc14967115)

[图 7 GAN生成的万能指纹，准确率高达76% 7](#_Toc14967116)

[图 8 2012-2014年利用行为生物测量学提高认证安全性的研究数量 8](#_Toc14967117)

[图 9 支付用户选择支付平台最关注的因素 12](#_Toc14967118)

[图 10 非网上支付用户不使用网上支付的原因 13](#_Toc14967119)

[图 11 中国网上交易额发展较快 14](#_Toc14967120)

[图 12 MVSA运行流程 15](#_Toc14967121)

[图 13 坐标轴示意图 17](#_Toc14967122)

[图 14 真用户数据和冒名顶替者数据的分布示意图 20](#_Toc14967123)

[图 15 MVSA系统运行示意图 23](#_Toc14967124)

[图 16 MVSA模块架构图 24](#_Toc14967125)

[图 17 MVSA密码输入示意图 25](#_Toc14967126)

[图 18 gaptime和touchTime示意图 26](#_Toc14967127)

[图 19 九宫格划屏偏移距离 26](#_Toc14967128)

[图 20 4位志愿用户样本数据触摸时间 33](#_Toc14967129)

[图 21 4位志愿用户样本数据触摸间隔 33](#_Toc14967130)

[图 22 4位志愿用户触屏时x轴最大角速度 34](#_Toc14967131)

[图 23 4位志愿用户触屏gapTime 34](#_Toc14967132)

[图 24 自动编码器 36](#_Toc14967133)

[图 25 认证结果显示 39](#_Toc14967134)

[图 26 志愿者1识别结果混淆矩阵 45](#_Toc14967135)

[图 27 志愿者2识别结果混淆矩阵 45](#_Toc14967136)

[图 28 用户多场景下数据测试结果 46](#_Toc14967137)

[图 29 同一个用户数据迭代时的重建误差变化 47](#_Toc14967138)

[图 30 真用户的数据和假数据在MAE上的分布情况 47](#_Toc14967139)

[图 31 真用户的数据和假数据在MSE上的分布情况 48](#_Toc14967140)

[图 32 MVSA系统下某测试用户ROC曲线 48](#_Toc14967141)

表目录

[表 1 按键密码采集的数据字段及含义 24](#_Toc14967103)

[表 2 九宫格密码的数据字段及含义 25](#_Toc14967104)

[表 3 MVSA按键密码二分类模型下测试结果 44](#_Toc14967105)

[表 4 MVSA系统按键密码无监督异常检测模型测试结果 49](#_Toc14967106)

[表 5 MVSA与现存相关工作性能对比 49](#_Toc14967107)

[表 6 MVSA系统九宫格密码二分类模型测试结果 50](#_Toc14967108)

[表 7 MVSA系统九宫格密码无监督异常检测模型测试结果 50](#_Toc14967109)

# 摘要

计算机、通信以及网络技术的进步正在将传统台式电脑的应用转移到移动设备。2016年，全球已有超过20亿的智能手机用户。对这些设备的日益依赖不可避免地带来了存储在这个平台上的敏感数据的增加。不幸的是，移动设备的便携性也使其容易受到盗窃。数据泄露和被盗设备的滥用可能比设备本身的成本更加令人担忧。由于密码本身的性质，一旦恶意攻击者通过某些特殊手段获取到合法持有者的密码，便可以拥有与之等同的权限来进行恶意操作，例如盗刷银行卡，支付宝等。基于上述原因，我们设计和实现了**一种基于机器学习的安卓多因素认证系统MVSA——The Multifactor Verification System for Android Based on Machine Learning。MVSA通过预先采集用户输入密码时传感器所收集到的数据流，为每个用户训练一个用户身份认证模型，当用户进行登录操作时，会用模型对登录时采集到的数据流进行判决，给出用户是否是真正用户的判决结果。**

在MVSA中，我们的创新性主要有：

**（1）多因素认证。**传统的密码验证方式如支付宝和微信的6位支付密码，只验证用户输入的密码是否正确，如果密码正确则能通过验证。传统的手机解锁密码有4位或6位密码，也有九宫格划屏解锁密码，均只对密码序列进行验证。一旦密码泄露，当用户暂离手机或手机丢失时，存在资金隐患。而我们的MVSA系统采用多因素认证机制，不仅仅局限于验证密码是否正确，只有当输入密码的行为与用户录入数据时一致，才能通过认证。

**（2）在安卓多因素认证系统这一案例中首次引入无监督异常检测模型。**以往所提出的分类方法多是在同时给定正负样本输入的基础之下进行分类器模型的训练，这类分类器虽然应用效果不错，然而具有较大的局限性。在本作品所使用的方法之中，我们在确保识别较高的识别正确率（93%以上）的前提之下，使用了无监督异常检测模型，限制条件更少，更贴近实用场景。

**（3）同时支持六位数字密码输入与九宫格划屏密码输入。**我们的系统提供了六位数字密码输入键盘与九宫格划屏密码输入两种认证模式。在最初所考虑的主流的六位支付密码场景中达到了较高准确率的基础上，我们将方法拓展到九宫格划屏密码场景之中，可以用来作为手机锁屏解锁密码。九宫格划屏密码能够提供更加全面的用户行为特征数据，实验结果表明该方法在九宫格划屏密码下有更高的识别准确率。

我们的实用性体现在：

**（1）用户操作无感。**MVSA与正常的密码认证操作相同，我们所做的认证系统隐藏了复杂的技术细节部分，在提高认证安全性的同时，不需要用户提供任何额外的操作。

**（2）用户使用便捷。**用户只有在初次使用MVSA系统或者重置密码时，需要预先进行数据录入。经过我们的实验，用户录入20~30个有效样本，MVSA系统就能够提供较高的认证准确率。通常情况下，用户录入这些样本只需要2~3分钟，在使用上是非常便捷的。

**（3）较低的用户资源开销。**MVSA系统拥有较低的用户资源开销。我们使用的传感器均是安卓提供的低功耗传感器，对手机的电量消耗比较低。我们的数据处理、模型训练模块是由服务端完成，所以在客户端上进行的数据运算量较小，不会给用户带来较大的资源开销。

**（4）用户在多场景输入下系统的稳定性与可靠性。**我们在设计MVSA系统时加入了对于用户输入场景的考量。由于手机这类移动设备的便携性，用户可能在多种场景下执行系统认证操作。因此，在实验测试阶段，我们邀请了志愿者在各种不同的场景下录取实验数据进行系统测试实验（场景包括但不限于站姿，坐姿，洗澡前后，是否乘坐交通工具等多个常见认证场景）。经过测试，我们的系统MVSA在多种场景下是稳定可靠的。我们还进行了抗模仿攻击测试，测试结果表明，在多种场景下，我们的系统拥有较高的准确率（93%以上），是稳定可靠的。

**关键词：多因素认证； 行为特征检测； 无监督异常检测**

# 作品概述

本章将从选题背景，国内外研究现状，特色描述以及市场分析四个部分对我们设计的MVSA系统进行概述。

## 选题背景

计算机、通信以及网络技术的进步正在将传统台式电脑的应用转移到移动设备。2016年，全球已有超过20亿的智能手机用户[1]。智能手机时代，人们对于手机的使用需求渐渐发生变化，从一开始的只关注手机的基本功能，逐步衍生出更多不同的需求。而随着需求的不断增加，手机也存储了越来越多的敏感信息[2]。



图 1 智能手机使用需求[3]

目前我国电子支付发展迅速，使用普及率较高，农村地区60%以上的成年人使用过电子支付，中国电子支付市场规模近3万亿美元[4]。调查[5]显示，在支付宝和微信的刺激和带动下，2016 年中国电子支付市场规模达到 2.9 万亿美元，在过去 4 年中增长了 20 倍。2017 年，全国使用电子支付成年人比例为76.9%，农村地区使用电子支付成年人比例为66.51%。非现金支付业务量平稳增长，移动支付业务量继续较快增长。2017年，全国共办理非现金支付业务1608.78亿笔，金额3759.94万亿元，同比分别增长28.59%和1.97%。手机信息已经渐渐地与银行账户结合在一起，涉及个人切身经济利益[6]。

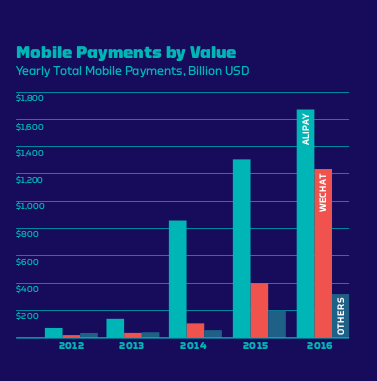


图 2 中国电子支付的快速增长[4]

不幸的是，移动设备的便携性也使其容易受到盗窃[7]。数据泄露和被盗设备的易被盗窃性可能比设备本身的成本更加令人担忧[8]。

如今，基于知识的认证方法，如数字密码、PINs码或图案锁(以下称为密码)，仍然是用于认证移动用户的主要方法[9]。然而，这些方法容易受到大量的攻击。此外，移动设备的使用模式通常是短时间的多次使用[10]，与台式计算机相比，这显著增加了所需的身份验证频率，更高的身份验证频率会降低验证系统的可用性[11]。为了平衡安全性和可用性之间的冲突，以前有部分文献提出使用延迟自动识别的方法，例如在需要重新验证请求之前，将设备设置为空闲时间[12]。目前关于密码认证，国内外研究者提出了很多相关方法，但是这些方法在提高安全性的同时，也给使用者带来了极大的不方便。中国互联网用户中大多数为非高知用户。作为全民级移动通讯工具，全国网民年龄、学历、收入结构调查显示2017年具有中等教育水平（初中至大专）、中等收入 （月收入 2001-5000 元）的群体仍是整体网民的主力军。尽管高知人群掌握互联网热点话题方向，但基数庞大的非高知人群则是中国网民的中流砥柱。这些占绝大多数比例的网民往往没有意识到手机上的敏感信息的安全重要性，因此往往也并不会去使用过于复杂的安全措施[5, 6, 13]。

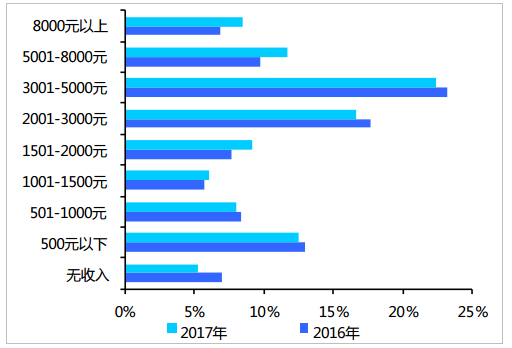


图 3 网民多集中在 2001-5000 元月收入区间

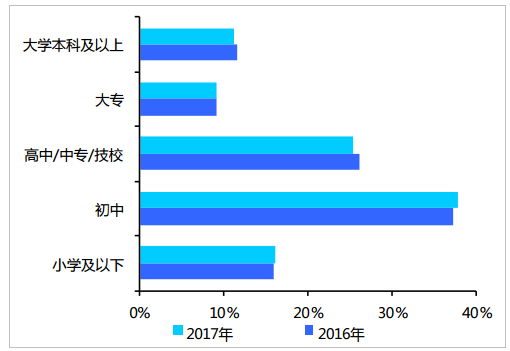


图 4 非高知类网民占据巨大多数比例

图 5 中国网民安全意识薄弱

尽管高知人群掌握互联网热点话题方向，但基数庞大的非高知人群始终是中国网民的中流砥柱。这些占绝大多数比例的网民却往往对手机上的敏感信息安全的重要性没有明确的认知，因此也并不会去使用过于复杂的安全措施。而移动支付应用中最为常用的支付宝所支持的部分安全措施，正是由于其相对复杂性，对于绝大多数网民的支付安全并没有起到很好的保护作用。



图 6 支付宝的部分安全设置

密码认证方法的一种常见替代方法是生物识别认证。该认证系统主要是根据一个人的生理或行为特征来识别他/她。生理生物识别技术是一种利用相对稳定的人体物理特征，如指纹、面部特征和虹膜图案，来进行个人身份鉴别的技术。由于生物密钥无法失窃或被遗忘，而且更难伪造[14]，因此生物特征识别认证方法被认为比其他认证方法(如密码)更安全、可靠、准确。

现有的基于生物特征的身份认证主要用到了指纹、声纹、虹膜、手势等生物特征。指纹识别具有全局信息和局部信息的唯一性，且人与人之间的差异较大，具有长期稳定性，因此指纹能够对人进行唯一标识，但是可能存在一些人群的指纹特征过少，不容易采集的情况。此外，攻击者也很容易对指纹进行复制。人脸识别是一种通过静态人脸图像和数据库中的人脸数据进行对比从而进行判断的身份认证方式，人脸识别技术使用便捷，在理想环境中准确率和安全性都能得到保证。但是，人脸匹配对人脸图像采集环境和手机硬件配置要求较高，对用户状态的要求也比较高。声纹与指纹类似，是能够表征人类身份的一种生物特征。声纹识别技术通过提取声纹这种语音特征参数，与声纹数据库中已有样本数据进行查找匹配，从而识别说话者的身份。但是由于声纹采集的非接触性，声纹信息非常容易被盗取和滥用。虹膜识别技术是基于眼睛中的虹膜进行身份识别，虹膜处于人类的巩膜中，以圆环状包围瞳孔。虹膜的纹理形成于胎儿时期，终生不会改变，且其纹理包含了200多种差异明显的特征。但是虹膜识别与人脸识别类似，对使用环境要求较高。手势识别则是通过采集用户手势的特征并对用户进行身份认证，具有使用方便，不会太占用资源等优点，但相关研究还不是很成熟。

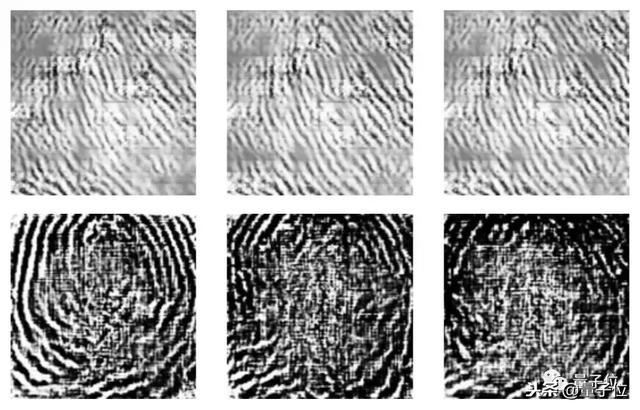


图 7 GAN生成的万能指纹，准确率高达76%

2015年的一次调查显示[15]，影响参与者选择在移动设备上使用哪种生物识别认证方法的重要因素之一是可用性。他们发现两个主要的可用性问题阻碍了参与者对生物特征鉴定方法的适应，一是鉴定速度慢且不方便，二是社交尴尬。例如，对于第一个问题，在人脸生物识别的情况下，参与者觉得在设备的摄像头前正确对齐人脸既困难又耗费时间。同时，在指纹生物识别之中，当手指太油或太干，或是当设备被保护套覆盖时，参与者将很难正确地扫描指纹。另一方面，至于第二个问题(即社交尴尬)，在公共区域持设备于人脸之前执行身份验证对于参与者来说是一件很尴尬的任务。

除去上文中所提到的种种认证方法，触控动力学则是一个解决该问题的全新思路。触控动力学是一种行为生物测量学，它捕捉一个人触摸触控设备的方式。与其他生物识别数据类似，触屏特征可用于识别个人/用户，也可与密码认证方法结合使用，以提高用户认证和移动设备保护的安全性。该方法可以通过在移动设备中嵌入现有的传感器来实现，因此相对于其他生物识别认证方法来说，它的成本较低。此外，这种方法是非侵入性的，可以与人的正常移动设备使用活动并行操作[16]。同时，该方法具有广泛的社会接受度，且触屏特征认证方法也有望被大众广泛接受[17]。由于触屏特征的潜力，这一领域的研究越来越多，如下图8所示。



图 8 2012-2014年利用行为生物测量学提高认证安全性的研究数量

本作品正是基于触控动力学这一思路，将用户击键特征与传统密码认证所结合起来的多因素密码认证系统。

## 研究现状

在过去的几年里，我们看到了大量关于击键生物识别的研究，而这其中的大多数研究论文都是关于计算机上的击键动力学以及一些综述，直到最近的一些论文之中才提及了对基于触摸屏的设备收集的数据集的评估。

Saevanee和Bhattarakosol[18]提出了第一个使用击键动力学和手指压力相结合的研究。通过从10个用户收集的数据集，他们证明仅使用手指压力信息就可以识别出99%的用户。识别使用的数据是用一个带有触摸屏的笔记本电脑收集的，参与者必须输入10位数字的手机号码。由于每个用户的电话号码不同，所以只能在数据集中测量FRR类型错误。对于远距误差测量，必须收集冒名顶替者的数据，因此该项研究的主要局限在于缺乏冒名顶用者的数据。另一项使用触摸屏特性的按键动力学研究是在Johansen的硕士论文中提出的[19]。这份研究的目的是比较个人电脑和智能手机的按键动态。一共有42人参加了该研究的相关实验，其中的一部分用户同时进行了个人电脑和智能手机的按键动态实验。这项研究的主要发现是，如若仅使用计时功能，智能手机的性能要比比标准键盘差。然而，如果使用除了计时功能之外的额外智能手机功能，其性能明显优于标准键盘。同时研究的结果还显示，在智能手机上模仿某人的打字节奏比在标准键盘上更难。此研究的主要局限在于数据收集过程中使用了一个12键手机键盘上的数字密码。Kang和Cho[20]研究了几种输入设备的击键动态认证系统:PC键盘、用手写笔输入的软键盘和触摸键盘。由于他们想要同时比较这些输入设备的性能，因此只能使用基于时间特性的击键特征。该研究采用高斯密度估计、Parzen窗密度估计、k近邻描述、支持向量数据描述等单类分类算法进行性能评价。最后的结果表明基于PC键盘的系统认证性能最好。Sen和Muralidharan[21]研究了使用基于触摸屏的移动设备输入4位PIN码的身份验证系统。该研究主要是在从10个用户收集的数据集上进行的，使用了压力和基于时间的特性进行识别。这一研究较前面的研究所不同的地方在于，除了进行身份验证测试之外，还对攻击场景进行了评估。

目前关于智能手机的相关研究比较少，而且更多的局限在数字键盘密码之上。Luca[22]、Orcan[23]分别提出了基于压力、速度和停留时间的多因素认证，但认证准确率不是很高，Luca的认证FRR为20%左右，Orcan的认证EER为7%左右。

本文专注于对手机身份认证这一场景进行了详细讨论，分析了各种情况下手机多因素认证的实现，给出了实用，准确，迅捷，安全实现方案。

## 特色描述

### 多因素认证

传统的支付密码验证方式，如支付宝和微信的6位支付密码，只验证用户输入的密码是否正确，如果密码正确则能通过验证。传统的手机解锁密码有4位或6位密码，也有九宫格划屏解锁密码，均只对密码序列进行验证，密码序列正确，则能够通过认证，解锁手机。一旦密码泄露，当用户暂离手机或手机丢失时，存在资金隐患。我们的MVSA系统采用多因素认证机制，不仅仅局限于验证密码是否正确，即便用户的手机丢失并且密码泄露，他人想通过输入正确密码来通过MVSA的认证机制的可能性是非常小的，从而保障用户的资金安全。

在系统实现上，我们用多维度全方位的特征来描述用户输入密码时的行为。一方面，我们利用安卓平台提供的内置传感器接口，记录下来用户输入密码时的触屏压力序列、触屏面积序列、触屏坐标序列、触屏间隔序列、手机加速度序列、陀螺仪传感器数据序列等数据作为特征；另一方面，我们在直接获取到的传感器数据基础之上，进行了用户输入数据低阶特征的提取，例如结合触屏坐标与触屏间隔序列计算划屏速度，提取每次交互中时间序列数据的最值和平均值等作为特征。在收集、计算得到所需特征后，将数据输入系统进行训练，得到用户的认证识别模型。以往所提出的该类问题解决方法往往只顾及到某一个角度或是特征，我们的系统使用全方位的特征来描述用户输入，从而保证了较高的准确率。

### 1.3.2 九宫格划屏密码输入键盘

我们的系统MVSA提供了两种认证场景，一种是支付场景，一种是手机解锁场景，分别对应于六位数字密码输入和九宫格划屏密码输入。最初，我们设想我们的系统是服务于支付密码的，为此我们针对六位支付密码进行了设计。在取得了较好的效果之后，我们将应用场景拓展到手机的锁屏密码中，针对九宫格划屏解锁密码进行了设计。九宫格划屏密码认证效果更好，能够提取到更加全面的用户行为特征数据，例如划屏速度、偏移距离等等，全面的数据特征更有利于对每个用户进行行为特征的建模，实验结果表明该方法在九宫格划屏密码下有更高的识别准确率。

### 1.3.3 在安卓多因素认证系统这一案例中首次引入无监督异常检测模型

以往所提出的身份认证分类方法多是二分类模型，即在同时给定正样本和负样本的基础之上进行分类器模型的训练，这类分类器虽然应用效果不错，然而在实际应用场景当中，往往只有正样本，并没有负样本，即只有真用户本人的数据，除非其他用户愿意提供自己的数据作为负样本。在MVSA的实现中，我们在确保识别较高的识别正确率（93%以上）的前提之下，使用了无监督异常检测模型，更贴近现实问题的处理模式，同时也能更好地保护用户的数据隐私。

### 1.3.4 用户操作无感

我们的MVSA系统简单易行，用户操作十分友好。MVSA与正常的密码认证操作完全相同，不需要用户提供任何额外的操作，系统隐藏了复杂的技术细节部分。用户与平常输入6位支付密码或划屏进行屏幕解锁的操作完全相同，却能够使用进行了安全增强的认证服务。

MVSA采用服务端-客户端模型搭建系统，经过实验测试验证，能准确、快速的检测出输入密码的人是否为真正的合法用户。测试结果表明，MVSA能在真实用户预先录入数据样本较少的情况下（20~30个正样本），对于未知输入数据，进行较为准确的判定（准确率93%以上）。MVSA系统能够在不影响用户体验的情况下设下安全保护网，拦截下恶意攻击者意欲利用正确密码窃取用户资金的行为。

### 1.3.5 用户使用便捷

用户初次使用MVSA系统或者重置密码时，需要预先进行数据录入。经过我们的实验，用户录入20~30个有效样本，MVSA系统就能够提供较高的认证准确率。通常情况下，用户录入这些样本只需要2~3分钟，在使用上是非常便捷的。

### 1.3.6 在多场景输入下系统的稳定性与可靠性

我们在设计MVSA系统时加入了对于用户输入场景的考量。由于手机这类移动设备的便携性，用户可能在，任何时间，任何地点执行系统认证操作。因此，在实验测试阶段，我们邀请了志愿者在各种不同的场景下录取实验数据进行系统测试实验（场景包括但不限于站姿，坐姿，洗澡前后，是否乘坐交通工具等多个常见认证场景），并以此为依据进行了系统的调整，确保MVSA系统在不同场景之下的的稳定实用性。

### 1.3.7 低资源开销

MVSA系统拥有较低的用户资源开销。我们使用的传感器均是安卓提供的低功耗传感器，对手机的电量消耗比较低。我们的数据处理、模型训练模块是由服务端完成，所以在客户端上进行的数据运算量较小，不会给用户带来较大的资源开销。

## 市场分析

保密性、完整性和可识别性是保障网上支付安全的三大因素。用户对支付安全性的诉求强烈，安全性与便捷性并驾齐驱成为用户最关注的因素。随着移动支付服务在网民生活中的加速渗透，方便和安全已经成为决定用户支付工具选择的核心因素。如下图9所示，调查中显示，用户选择支付平台最关注的因素是支付过程便捷性和支付安全性，分别有57.7%和51.5%的选择比例，远远超过排在第三、四位的有折扣或积分活动、品牌形象[6]。

图 9 支付用户选择支付平台最关注的因素

调查显示，安全担忧成为制约非网上支付用户选择使用网上支付工具的重要障碍。在不使用网上支付的网民中，有43.5%的网民是因为不需要网上支付工具，这部分人群的网络生活还没有从娱乐过渡到商务阶段，还处于网络应用的较浅层。而有30.4%的用户是因为感觉不安全、担心资金被盗而不使用网上支付，另有11.8%的用户担心账户信息泄漏。总的来看，这部分将近42.2%的人群出于安全性的考虑，尤其是资金和信息安全的担忧，而选择不使用网上支付，网上支付工具亟待消除人群对于安全性的顾虑。如果能够让用户相信网上支付是安全的，不会带来较大的资金风险，会在一定程度上激发网上支付用户的增长，促进电子商务行业的稳健发展。

图 10 非网上支付用户不使用网上支付的原因

自21世纪以来，网络零售和本地消费服务行业业务激增，随着中国网络零售市场的迅猛发展，线上消费的生活服务类型不断拓宽，交易规模持续增大，移动支付这一领域蕴含了巨大的市场空间。如下图11所示，在2007-2010年，中国网络零售市场交易规模增长近10倍，从542亿元增长到5231亿元人民币。2011年依然保持44.6%的增长幅度，全年交易规模达到7566亿元，网络零售市场交易总额占社会消费品零售总额的已经达到4.2%。2015年，我国网络零售市场已达到2万亿元人民币以上，超过美国；人均网上消费额将达到6220元人民币，超过美国目前1000美元的平均水平。网络零售成为新业态，电子商务交易额迅速上升，促进了网上支付行业的发展。

图 11 中国网上交易额发展较快

但就发展趋势来看，我国网上金融的增长还远远没有触顶，尤其是对于将成为未来网民增长重要群体的中年人群而言，还有着较大的渗透空间。从未来发展的预期看，我国互联网渗透逐步加深的势头不可逆转，网络消费供需面持续积极向好，这些都将推动网上银行、网上支付的发展，促进电子商务在未来较长时间稳健增长。

因此，网上支付安全性的提高刻不容缓。文中提出的MVSA系统能够实现便捷而有效的用户信息安全保护，在极大程度上满足用户对于支付信息安全保障的需求，且显著提高了用户支付行为中的安全性，保障了中国网上金融的稳健发展，具有极大的市场前景。

## 本章小结

本章从选题背景，国内外研究现状，特色描述以及市场分析四个部分着手，层层深入，抽丝剥茧对我们设计的MVSA系统进行一个系统性的概述。在下文之中，第二将详细的描述MVSA系统功能和模块的设计原理以及代码实现，第三章则是完整的阐述了我们在设计MVSA系统过程之中所进行的实验测试和结果分析。第四章是创新性说明，第五章则是对我们所做的工作进行的汇总与总结与展望。

# 第二章 作品设计与实现

本章将介绍MVSA系统功能，实现原理，所依托硬件设备，以及软件算法模块实现等内容。

## 2.1 系统功能

如上文所述，本作品旨在为用户在使用安卓手机输入密码时提供身份认证服务。用户使用MVSA系统时，首先通过收集用户客户端APP录入个人真实的数据信息，接着个人信息将会被发送至服务器端，服务器端为用户训练识别模型并存储。当下一次用户客户端使用APP输入密码时，输入数据发送至服务器端，先判断密码输入是否正确，如果正确，则用训练好的模型判断输入该密码的是否是该用户本人，将结果反馈回客户端。



图 12 MVSA运行流程

在MVSA系统的识别过程之中，以往所提出的分类方法多是在同时给定正负样本输入的基础上进行分类器模型的训练，这种方法虽然效果不错，却具有较大的局限性。而MVSA系统在确保较高的识别准确率（93%以上）的前提之下，使用了无监督异常检测模型。用户可以选择系统使用传统的二分类模型还是无监督异常检测模型，如果使用二分类模型，其数据也会在之后被用作其它用户训练时的负样本；如果使用无监督异常检测模型，则在训练时只会使用用户的数据进行训练，用户的数据不会作为其它用户训练时的负样本，这一模型限制条件更少，同时也能更好的保护用户的隐私。

## 2.2 系统依托的安卓传感器介绍

### （1）安卓基础传感器

Android系统提供了传感器的接口[24]，如果安卓设备的硬件内置了这些传感器，那么我们就可以通过API方便地获取手机传感器的数据。比如手机摆放环境外界的磁场、温度和压力、光照强度等等。

常用的安卓基础传感器有：

* 加速度计
* 环境温度传感器
* 磁场传感器
* 陀螺仪
* 心率传感器
* 光线传感器
* 近程传感器
* 压力传感器
* 相对湿度传感器

常用的安卓复合传感器有：

* 线性加速器
* 大幅度动作传感器
* 步测器
* 倾斜检测器
* 方向传感器

我们主要选取的是IMU模块的传感器，包括三轴线性加速度传感器和陀螺仪，采样时间间隔为10ms。其中三轴线性加速度传感器和陀螺仪的三个传感器坐标轴的方向，由设备决定，仅与屏幕的自然方向相关，与设备姿态无关。当设备的所处状态更改时，传感器的坐标轴不会发生交换。坐标轴示意图如下图13所示：

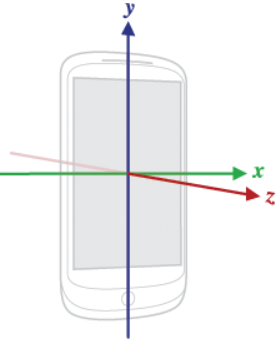


图 13 坐标轴示意图

线性加速度是线性加速度传感器可报告传感器框架内设备的线性加速度（不包括重力加速度）。从概念上看，输出结果为：加速度计的输出结果减去重力传感器的输出结果，单位为 m/s2。

当设备不动时，所有轴上的读数应接近 0。其返回的三个加速度说明如下：

* 第一个值：代表该设备沿X轴的加速度。
* 第二个值：代表该设备沿Y轴的加速度。
* 第三个值：代表该设备沿Z轴的加速度。

陀螺仪用于感应手机的旋转速度，给我们返回了当前设备的X、Y、Z三个坐标轴的旋转速度。旋转速度的单位是弧度/秒，正值代表逆时针旋转，负值代表顺时针旋转。关于返回的三个角速度说明如下：

* 第一个值：代表该设备绕X轴旋转的角速度。
* 第二个值：代表该设备绕Y轴旋转的角速度。
* 第三个值：代表该设备绕Z轴旋转的角速度。

### （2）安卓触屏接口

Android 支持各种触摸屏和触摸板，提供了触屏事件的数据接口，使得我们可以获得触屏事件发生时的一些数据，例如用户触屏坐标、触屏压力、触屏面积和触屏时间等，安卓的触屏事件数据主要包括以下几种：

* X 和 Y 字段：接触区域中心的位置信息。
* TouchMajor、TouchMinor、ToolMajor、ToolMinor、Size 字段：其中Size字段描述了相对于触摸设备可以感知的最大可能触摸区域的标准化触摸区域尺寸。可能的最小标准化尺寸为 0.0（无接触或不可测量），可能的最大标准化尺寸为 1.0（传感器区域已经完全覆盖）。
* Pressure 字段： 该字段描述了以介于 0.0（无接触）和 1.0（全力）之间的标准化值形式施加到触摸设备的近似物理压力。

## 2.3 身份认证算法和工具

MVSA系统的核心是基于用户输入密码行为特征与传统密码验证机制相结合的身份认证，可以看成一个二分类或者异常值检测问题。如果看成是二分类问题，即区分出当前输入正确用户密码的人是/不是真的用户本人。如果看成是无监督异常侦测问题，则通过当前的密码输入与用户数据库中信息比对，判断其是否是真用户输入。二者的区别是，二分类在训练时有正样本（用户本人数据）和负样本（随机匿名抽取的其它非该用户的数据），而无监督异常侦测则在训练时只使用正样本，更好的保护了用户的隐私。传统的分类检测问题更多的都是使用二分类方法，我们的创新点之一即MVSA系统是首个引入无监督异常侦测模型解决行为特征识别问题的系统。

二分类常用的分类器有[25]：

* 朴素贝叶斯
* K-近邻
* 支持向量机
* 决策树
* 神经网络
* 集成分类器（Gradient boost、Adaboost等）

无监督异常侦测常用的方法有：

* one-class SVM
* Isolation Forest
* Elliptic Envelope
* SVDD
* Kernel Density Estimation
* Gaussian Mixture Model
* Auto Encoder

在样本数据非常少的情况下，如本系统在录入时，不可能要求每个用户输入几百个数据，采用传统的神经网络的方法训练二分类模型，并不能取得很好的效果，传统的机器学习方法能取得更可靠和稳定的性能。k近邻、决策树、Gradient Boost和Adaboost在测试数据集上的性能不是很稳定，经过前期验证，采用SVM的方法，通过精心构造、挑选特征，结合概率阈值，能取得较好的判决效果。

在无监督异常侦测上，MVSA系统选择使用自动编码器，通过对重建误差设置阈值，能取得比较好的效果。我们在无监督异常侦测的前期实验上，使用过GMM模型，在我们的数据集上，通过固定阈值甚至能取得比二分类模型更好的效果，TPR接近98%，FPR接近0.5%，但是我们没有采用，因为在实际应用场景中GMM的固定阈值没有办法很好的设置下来贴合实际应用。所以，出于实用性的考量，MVSA的二分类模型使用的是支持向量机模型，无监督异常侦测场景使用的是自动编码器模型。

对支持向量机二分类模型进行训练，或者只使用正样本，利用无监督异常侦测的方法进行训练，目的是对于一个新样本，判断其来自真用户，还是冒名顶替的用户。判断方法是：若是二分类模型，通过模型，对新来的样本计算一个信心分数，如果分类器的信心分数高于阈值，认为其来自真用户，是正常的样本，否则，就认为是异常的，来自冒名顶替者；若是无监督异常侦测模型，通过模型，对新来的样本进行数据重建，之后计算重建误差，如果重建误差比较小，认为其来自真用户，是正常的样本，否则，就认为是异常的，来自冒名顶替者。

在二维的情况下，假设真用户数据和冒名顶替者数据的分布示意图如图14所示。二维情况下，真的用户的样本通常会比较集中，假设其大致分布在一个圆形区域内，而经过训练之后，分类器的某概率下的判决边界近似是一个圆。大部分情况下，如果阈值选择合理，在这种概率下，异常用户的数据会被判为异常点，真用户的数据会被接受为正常点；但是当异常点和正常点非常接近，难以分辨时，会发生判决错误。如果正常点太过偏离正常点的中心，也可能会出现正常点被误判为异常点的情况。我们通过设置合理的阈值，选择一个合理的判决边界，能够尽可能地将真用户的数据接受为正常点，将异常用户的数据判为异常点，达到较好的分类效果。

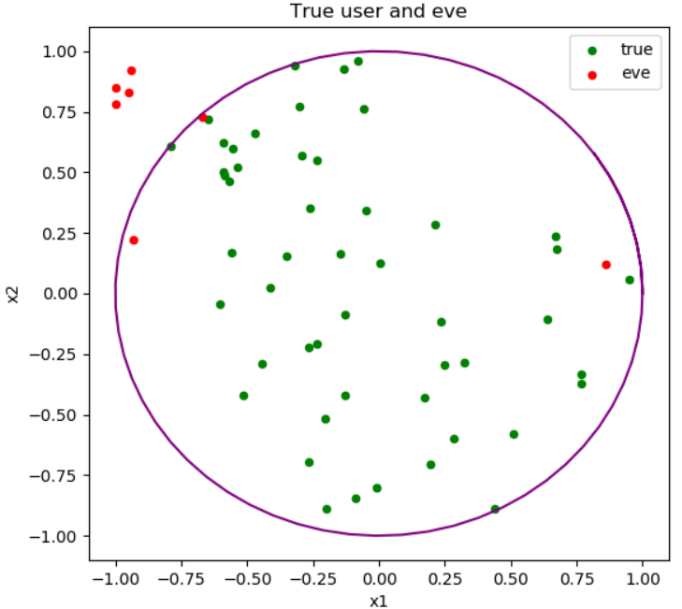


图 14 真用户数据和冒名顶替者数据的分布示意图

### （1）支持向量机

支持向量机是一种二分类模型，它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使得它有别于感知机，支持向量机还包括核技巧，这使它成为实质上的非线性分类器，支持向量机的学习策略就是间隔最大化，可形式化为一个求解凸二次规划的问题，也等价于正则化的合页损失函数的最小值问题。支持向量机的学习算法是求解凸二次规划的最优化算法。

支持向量机学习方法包含构建由简至繁的模型：线性可分支持向量机、线性支持向量机及非线性支持向量机。当训练数据线性可分时，通过硬间隔最大化，学习一个线性的分类器，又称为硬间隔支持向量机；当训练数据近似线性可分时，通过软间隔最大化，也学习一个线性的分类器，即线性支持向量机，又称为软间隔支持向量机；当训练的数据线性不可分时，通过核技巧及软间隔最大化，学习非线性支持向量机，核函数往往取高斯径向基函数。

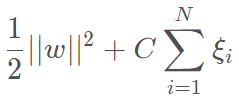
假设给定一个特征空间上的训练数据集：

T = {(x1,y1),(x2,y2),...,(xn,yn)}

其中，xi∈***X*** = **R**n，yi∈***Y*** ={+1,-1}，i=1,2,...,n,xi为第i个特征向量，yi为xi的类标记，假设训练数据集不是线性可分的，线性不可分意味着某些样本点(xi,yi)不能满足线性可分支持向量机的函数间隔大于等于1的约束条件。为了解决这个问题，可以对每个样本点引入一个松弛变量*ξi*​≥0，使得函数间隔加上松弛变量大于等于1，这样，约束条件为：

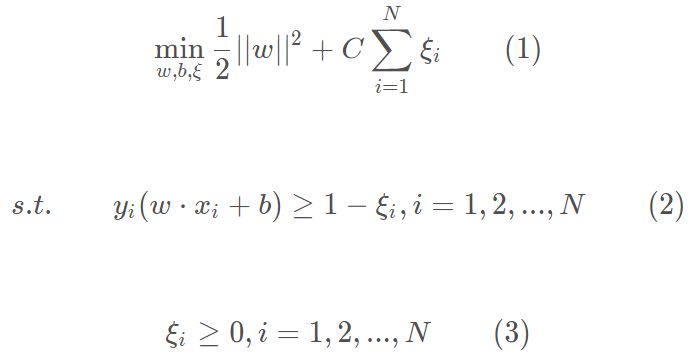


同时，对每个松弛变量*ξi*，支付一个代价*ξi*，目标函数为：



这里，C>0称为惩罚函数，在训练中，我们选择C=1。C值大时对误分类的惩罚增大，C值小时对误分类的惩罚减小。

线性不可分的支持向量机的学习问题为如下凸二次规划问题：



原始问题(1)～(3)是一个凸二次规划问题，因而关于(w,b,ξ)的解是存在的，可以证明w的解是唯一的，但b的解可能不唯一，而是存在于一个区间。

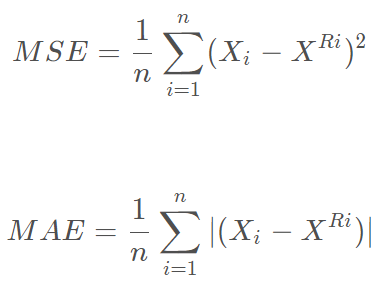
设问题(1)～(3)的解是w\*，b\*，于是可以得到分离超平面w\*·x + b\* = 0及分类决策函数f(x) = sign(w\*·x + b\*)。比起线性可分支持向量机，线性支持向量机具有更广的适用性。

### （2）无监督异常侦测之——自动编码器

自动编码器是一种无监督的神经网络模型，它可以学习到输入数据的隐含特征，这称为编码，同时用学习到的新特征可以重构出原始输入数据，称之为解码。从直观上来看，自动编码器可以用于特征降维，类似主成分分析PCA，但是其相比PCA其性能更强，这是由于神经网络模型可以提取更有效的新特征。除了进行特征降维，自动编码器学习到的新特征可以送入有监督学习模型中，所以自动编码器可以起到特征提取器的作用[26]。

异常侦测通常分为有监督和无监督两种情形。在无监督的情况下，我们没有异常样本用来学习，而算法的基本上假设是异常点服从不同的分布。根据正常数据训练出来的自动编码器，能够将正常样本重建还原，但是却无法将异于正常分布的数据点较好地还原，导致还原误差较大。如果样本的特征都是数值变量，我们可以用MSE或者MAE作为还原误差。

如果输入样本为X=(X1,X2,⋯,Xn)，经过自动编码器重建的结果为XR=(XR1,XR2,⋯,XRn)，则还原的均方误差MSE和平均绝对差MAE为：



当还原误差大于某个阈值时，我们将其标记为非真正用户，从而实现异常侦测。

## 2.4 MVSA系统的运行流程

MVSA系统的运行流程如图15所示，用户使用本系统，首先需要在手机端使用APP输入密码录入自己真实的数据样本，经过实验，需录入20~30个左右的有效样本才能够比较有效地进行身份认证，且不会让用户在录入样本时产生抵触心理（实验时录入了30个有效样本，实际上只需要20个样本进行训练即可，另外10个样本主要是用于实验时的测试）。当录入完毕后，真实用户的样本数据会被传送到MVSA服务器端后，在服务器端进行数据处理，为该用户训练模型，并进行存储。

当用户需要执行登录操作时，用户在手机端输入密码，数据传送至服务器端，服务器端调用之前为用户训练好的模型判断现在输入密码的是否是真正的用户，并返回判断结果。如果的确是用户本人输入的密码，有很大几率会判断为是真用户操作；如果是冒名顶替者，其偷窃或用其它方式获得了真正用户的手机及其密码，在使用本系统进行登录时，虽然通过了密码部分的认证，但是因为其输入密码的行为特征与真正的用户差别较大，所以有很大几率会判断为非真正用户，从而能够有效地保障用户的数据和支付安全。

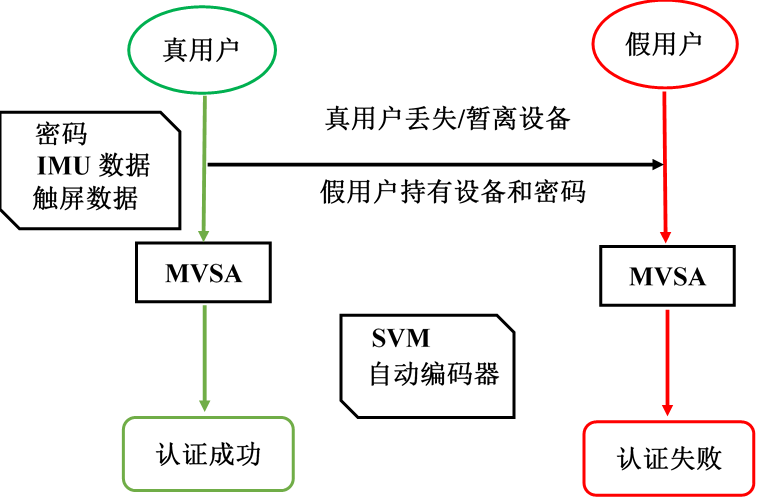


图 15 MVSA系统运行示意图

MVSA系统同时支持按键密码与九宫格密码，以及使用二分类模型或无监督异常检测模型。如果使用二分类模型，其数据传到服务器上后，有可能会被作为匿名的数据，在其它用户训练时，作为训练集上的负样本；如果使用无监督异常检测模型，则该用户模型在训练时只使用正样本，其数据也不会在其他用户的模型训练阶段进行匿名共享。我们更加推荐使用无监督异常检测模型这一识别模式，在我们所提出的MVSA系统之中，无监督异常检测模型显然可以更好的保护用户的隐私。

## 2.5 MVSA系统模块介绍

MVSA系统主要分为以下三大模块：用户数据采集模块、用户数据训练模块、以及用户验证登陆模块。模块架构图如下图16所示。下面的小节将分别对MVSA系统的每个模块进行详细的介绍和分析。

### 2.5.1用户数据采集模块

#### （1）数据采集内容

系统需要采集用户输入密码时的以下信息：

1. 密码

2. 触屏数据（包括触摸压力大小、触摸时间、触摸间隔、触摸中心坐标、触摸面积）

3. 线性加速度传感器和陀螺仪数据

4. 九宫格的触摸坐标偏移距离



图 16 MVSA模块架构图

按键密码采集的数据字段及含义，如下表所示:

表 1 按键密码采集的数据字段及含义

|  |  |
| --- | --- |
| **数据项名称** | **数据项含义** |
| Xord | 触屏时的X坐标 |
| Yord | 触屏时的Y坐标 |
| Pressure | 触屏时压力 |
| Presize | 触屏时触摸面积 |
| Touchtime | 开始一次触摸到结束该次触摸的时间 |
| Gaptime | 本次触摸距离上一次触摸的间隔时间 |
| ACC | 手机线性加速度传感器的数据 |
| DIR | 手机陀螺仪的数据 |

表 2 九宫格密码的数据字段及含义

|  |  |
| --- | --- |
| **数据项名称** | **数据项含义** |
| Xord | 触屏时的X 坐标 |
| Yord | 触屏时的Y坐标 |
| Pressure | 触屏时压力 |
| Presize | 触屏时触摸面积 |
| distance | 触摸坐标偏移距离 |
| speed | 划屏速度 |
| Gaptime | 从一个按钮划到下一个按钮耗时 |
| ACC | 手机线性加速度传感器的数据 |
| DIR | 手机陀螺仪的数据 |

数据采集中需要特别说明的部分包括密码、触摸时间和间隔及九宫格密码的触摸坐标偏移距离：

1. **密码**

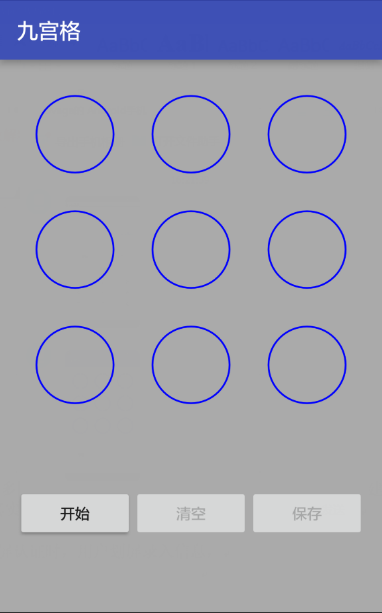


图 17 MVSA密码输入示意图

按键密码多因素认证时，我们的系统暂时设置6位数的密码，建议每个用户录入30个以上的真实数据。用户录入信息时，会记录其录入的密码。点击开始，录入信息，其中，因为在前期试用时，我们发现，用户在点击开始后，习惯停一会儿，所以点击开始的这次点击信息，不计为用户信息。

九宫格密码中，用户划屏录入信息，从上至下密码为[1 2 3 4 5 6 7 8 9]，用户可以划屏输入密码如“1236987”，然后单击保存按钮。

1. **触摸时间和间隔**

在按键密码中，我们考虑触摸时间和间隔，触摸时间touchTime是从手指触摸到按键，直到抬起手机的时间；而gapTime是从上一次触摸按键抬起手指，到下一次触摸的时间间隔。

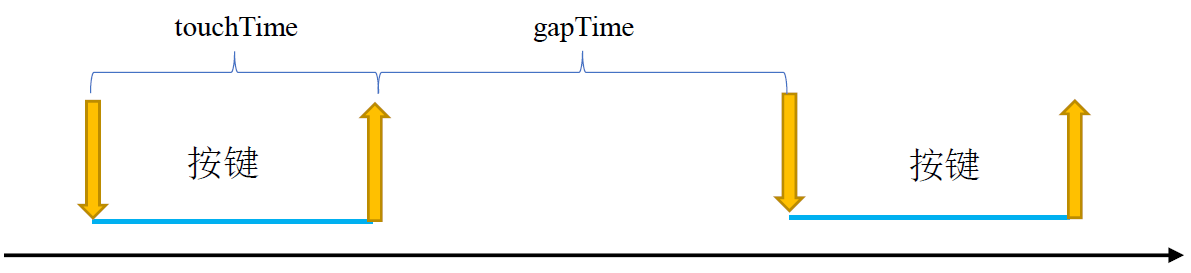


图 18 gaptime和touchTime示意图

1. **九宫格触摸偏移距离**

在九宫格密码中，我们定义了一个特征——偏移距离，在数据采集过程中进行计算。在划动过程中，假设密码是7位，从第1位划到第2位的过程中，我们定义直线划动位移是从第1个按钮中心直线移动到第2个按钮的中心，密码为“1236987”时全部为直线划动位移的情况如图19所示。划动过程中，每个点的坐标与这两个中心连线的距离，定义为该坐标点的偏移距离。

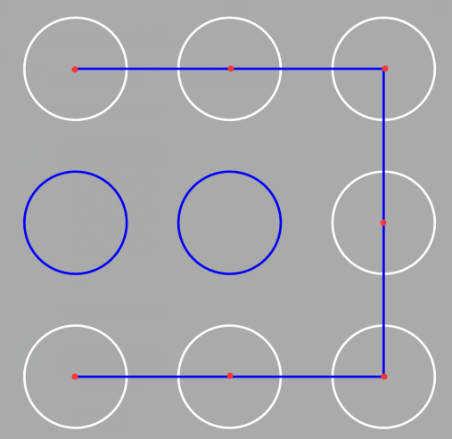


图 19 九宫格划屏偏移距离

#### （2）数据采集模块实现

1. **触屏数据**

获取触屏数据，利用Android的MotionEvent获得。当用户第一次触摸屏幕时，会产生一个ACTION\_DOWN事件，触摸的坐标、力度、面积数据也会产生；当用户的手指继续在屏幕上划动时，会产生ACTION\_MOVE事件，数据会继续更新；当用户抬起手指时，会产生ACTION\_UP事件，这一次触摸结束。

通过getAction()获取当前发生的屏幕事件，通过getX()和getY()获得触摸事件的屏幕坐标，通过getSize()获得触摸面积，通过getPressure()获取触摸压力。而触摸时间touchTime则通过计算从ACTION\_DOWN状态到ACTION\_UP状态的差值获得，触摸间隔时间gapTime通过计算这一次触摸的ACTION\_DOWN事件距离上一次触摸的ACTION\_UP时间的时间差来获得。

完整的触屏数据获取代码如下：

|  |
| --- |
| public boolean onTouch(View v, final MotionEvent event) {  switch (event.getAction()) {  //点击的开始位置  case MotionEvent.ACTION\_DOWN:  if(processState==0)//如果没有进入采样模式  {  break;  }  touching=1;  firsttime++;  if(username.equals("")) {  firsttime=0;  if(!AccList.isEmpty())  {  clean();  }  Toast.makeText(context, "用户id不能为空",Toast.LENGTH\_SHORT).show();  }  else {  touchTime = System.currentTimeMillis();//获取当前时间  times++;  SeqList.add(times);  AccList.add((float)-6);AccList.add((float)-6);AccList.add((float)-6);  GyrList.add((float)-6);GyrList.add((float)-6);GyrList.add((float)-6);    if(firsttime==1)//第一次按键  {  times=0;  //使用Timer采样IMU数据  if(timer==null) {timer=new Timer();}  timer.schedule(new TimerTask()  {  @Override  public void run() {  times++;  SeqList.add(times); AccList.add(AccData[0]);AccList.add(AccData[1]);  AccList.add(AccData[2]);GyrList.add(GyrData[0]); GyrList.add(GyrData[1]);GyrList.add(GyrData[2]);  }  },0,SENSOR\_RATE\_FASTEST);  }  gapTime=System.currentTimeMillis()-gapTime;//计算间隔时间  Gaptime.add(gapTime);  sampletimes++;  Xord.add(event.getX());//X坐标  Yord.add(event.getY());//Y坐标  Pressure.add(event.getPressure());//压力  Presize.add(event.getSize());//面积  inputnums.setText("当前输入了 "+ String.valueOf(firsttime)+" 个数字");  }  break;  //触屏移动实时位置  case MotionEvent.ACTION\_MOVE:  sampletimes++;  Xord.add(event.getX());  Yord.add(event.getY());  Pressure.add(event.getPressure());  Presize.add(event.getSize());  break;  //离开屏幕的位置  case MotionEvent.ACTION\_CANCEL:  case MotionEvent.ACTION\_UP:  touching=0;  touchTime = System.currentTimeMillis()- touchTime;//通过差值获得触屏时间  gapTime=System.currentTimeMillis();    //完成记录  Touchtime.add(touchTime);  for(int i=0;i<sampletimes-1;i++)  {  Touchtime.add((long)-1);  Gaptime.add((long)-1);  }    /\*写入分界线-2.0\*/  Xord.add((float) -2);  Yord.add((float) -2);  Pressure.add((float) -2);  Presize.add((float) -2);  Touchtime.add((long)-2);  Gaptime.add((long)-2);  times++;  SeqList.add(times);  AccList.add((float)-2);AccList.add((float)-2);AccList.add((float)-2);  GyrList.add((float)-2);GyrList.add((float)-2);GyrList.add((float)-2);  sampletimes=0;  //如果已经按下了6个键  if(firsttime==6) {  timer.cancel();  timer=null;  writetoxls();  clean();  input\_begin.setEnabled(true);//可以再次开始采集  input\_clear.setEnabled(false);//开始输入了，不能再开始了  }  default:  break;  }  return true;//继续响应触屏事件  } |

1. **IMU数据获取**

首先，我们要通过getSystemService获取传感器对象：

|  |
| --- |
| sensorManager=(SensorManager)context.getSystemService(Context.SENSOR\_SERVICE; |

接着，我们在onResume()中获取传感器的类型，并注册传感器，本系统中，我们选择使用线性加速度传感器和陀螺仪的数据。每次采样开始前会调用onResume()注册传感器，而在一次采样结束后调用onStop()注销传感器。

|  |
| --- |
| // TYPE\_LINEAR\_ACCELERATION是线性加速度传感器  // TYPE\_LINEAR\_ACCELERATION是陀螺仪传感器  // SENSOR\_DELAY\_FASTEST采用最小延迟采样  //注册传感器  protected void onResume() {  if(processState==0)  {  sensorManager.registerListener(this, sensorManager.getDefaultSensor(Sensor.TYPE\_LINEAR\_ACCELERATION),  SensorManager.SENSOR\_DELAY\_FASTEST);  sensorManager.registerListener(this,  sensorManager.getDefaultSensor(Sensor.TYPE\_GYROSCOPE),  SensorManager.SENSOR\_DELAY\_FASTEST);  processState=1;  }  }  //注销传感器  protected void onStop() {  if(processState==1) {  processState=0;  sensorManager.unregisterListener(this);  }  } |

在注册了传感器后，当传感器的数据发生了变化时，会触发传感器事件，我们在其中进行数据的采集，首先判断触发事件的是何种传感器，然后将这次传感器事件的数据event.values保存在对应的变量中。

|  |
| --- |
| public void onSensorChanged(SensorEvent event) {  int type = event.sensor.getType();//获取触发事件的传感器种类  float[] values = event.values; //获取这次传感器事件的数据  if(processState==1)//如果正在触摸  {  switch(type) {  case Sensor.TYPE\_LINEAR\_ACCELERATION://如果是线性加速度传感器  AccData[0]=values[0];AccData[1]=values[1];AccData[2]=values[2];  break;  case Sensor.TYPE\_GYROSCOPE://如果是陀螺仪  GyrData[0]=values[0]; GyrData[1]=values[1]; GyrData[2]=values[2];  break;  default:  break;  }  }  } |

1. **九宫格划屏偏移距离**

九宫格密码中，因为我们主要做的是多因素认证，所以假设用户会输入正确的密码，如果划错了，没有办法通过第一道认证。在划动过程中，假设密码是7位，从第1位划到第2位的过程中，我们定义直线划动位移是从第1个按钮中心直线移动到第2个按钮的中心，而划动过程中，每个点的坐标与这两个中心连线的距离，定义为偏移距离。偏移距离的计算方式是：

|  |
| --- |
| 设前一个密码键中心位置坐标为（x1,y1），下一个密码键中心位置为（x2,y2），当前用户处于从前一个密码键划动到下一个密码键的过程中，此时触摸坐标为(xt,yt)，那么有：  1)若x1=x2，则距离d= | xt - x1 |；  2)若y1=y2，则距离d= | yt - y1 |；  3)若x1≠x2且y1≠y2，计算连接两点的直线为y = kx + b，其中k=(y2-y1)/(x2-x1)，b = y1-kx1，距离d= |kxt - yt + b| / sqrt(k2+1) |

代码实现如下：

|  |
| --- |
| if(recode\_state==1)//如果还在划动  {  int i = Integer.valueOf((pass.substring(pass.length() - 1))).intValue() - 1;//目前的按钮  char ch = MainActivity.password.charAt(pass.length());  int j = (int) ch - (int) '0';  int now\_x1center=keys[i].bound.left+keys[i].bound.right;  int now\_y1center=keys[i].bound.top+keys[i].bound.bottom;  int now\_x2center=keys[j-1].bound.left+keys[j-1].bound.right;  int now\_y2center=keys[j-1].bound.top+keys[j-1].bound.bottom;  double x1center=(double)now\_x1center/2;  double y1center=(double)now\_y1center/2;  double k=0;  double b=0;  if(now\_x1center==now\_x2center)//斜率不存在，竖着的，距离就是|xn - x1ccenter|  {  now\_distance=(double)Math.abs(event.getX()-x1center);  }  else if(now\_y1center==now\_y2center)//如果是水平的  {  now\_distance=(double)Math.abs(event.getY()-y1center);  }  else//点斜式计算距离  {  k=(((double)now\_y1center/2)-  ((double)now\_y2center/2))/(((double)now\_x1center/2)-((double)now\_x2center/2));  b=(double)now\_y1center/2 - k\*(double)now\_x1center/2;// b=y1-kx1  now\_distance=Math.abs(k\*event.getX()-event.getY()+b)/Math.sqrt(k\*k+1);  }  Distance.add((float)now\_distance);//距离  } |

### 2.5.2用户数据训练模块

#### （1）训练模块算法介绍

1. **特征选取和预处理**

对于数据采集模块中采集到的大量的输入密码过程中的数据，我们需要先选取最有效、最稳健的部分特征。已有的文献中，较为常用的特征包括触摸时间和间隔、压力、面积等，我们认为，在触摸过程中，同样有价值的还有触摸坐标和IMU数据。对于获得到的数据，在尝试中，MVSA系统使用过DTW和LSTM等时间序列分类算法，效果并不是特别好，所以我们把数据进行规整处理，选取其在一次触摸过程中的最大、最小和均值。

值得注意的是，在用户数据采集模块收集到数据之后，我们发现，虽然Android的API中给出了获取压力和面积的接口，但是并不是每一款手机都支持获取压力和面积，如果手机不支持压力或者面积，该项采集到的数据就会是0或者1。大多数手机只支持压力或面积中的一个，所以在具体实验中，MVSA系统会依据具体情况，在训练的过程之中选择二者中的一个。

在进行大量的建模测试后，对于用户触摸输入密码的特征，我们最终选定了坐标、压力（面积）、触摸时间、触摸间隔、线性加速度作为最稳健的用户触摸特征，这些特征能有效地将真用户和其它用户区分开来，而用户在输入按键密码时，陀螺仪的数据读数本身较小，所以就不作为训练特征。

在随机选取的4位用户的多个样本中，输入6位密码分别的触摸时间如下图20所示，同一用户的分布基本上较为集中，不同用户之间有一定的区分度。他们的触摸间隔则如下图21所示。

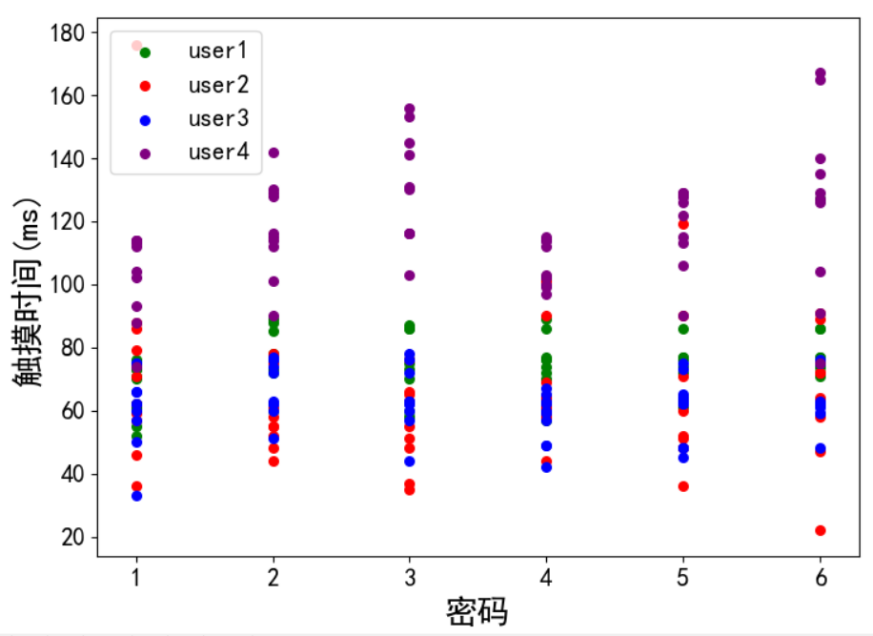


图 20 4位志愿用户样本数据触摸时间

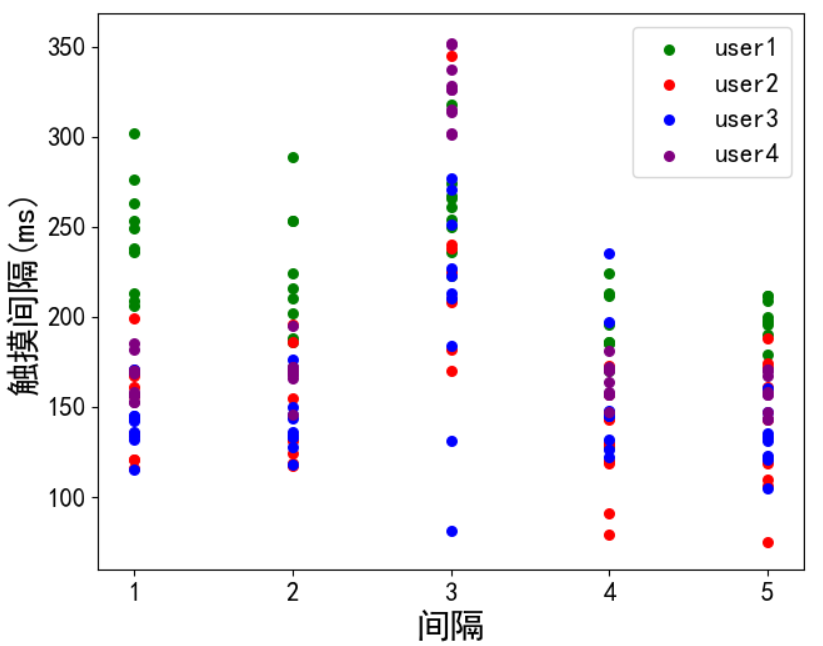


图 21 4位志愿用户样本数据触摸间隔

同样的，对于九宫格密码，MVSA系统将坐标、压力（面积）、速度、偏移距离、陀螺仪数据作为用户的九宫格密码的用户触摸特征，这些特征能有效地把真用户和其它用户区分开来，而线性加速度读数因为在划屏过程中相较于陀螺仪，较为随机，所以没有选取其为一个特征。其中速度特征需要在数据处理阶段进行计算。我们假设用户从一个坐标点划到下一个坐标点时，这短短的时间内，是匀速直线运动，通过位移和时间，我们可以计算这个过程中的速度。九宫格密码的4个用户样本的x轴最大角速度图22所示。而这个过程中，从一个键划到另一个键的耗时gapTime如图23所示，4个用户的区分度也比较大。

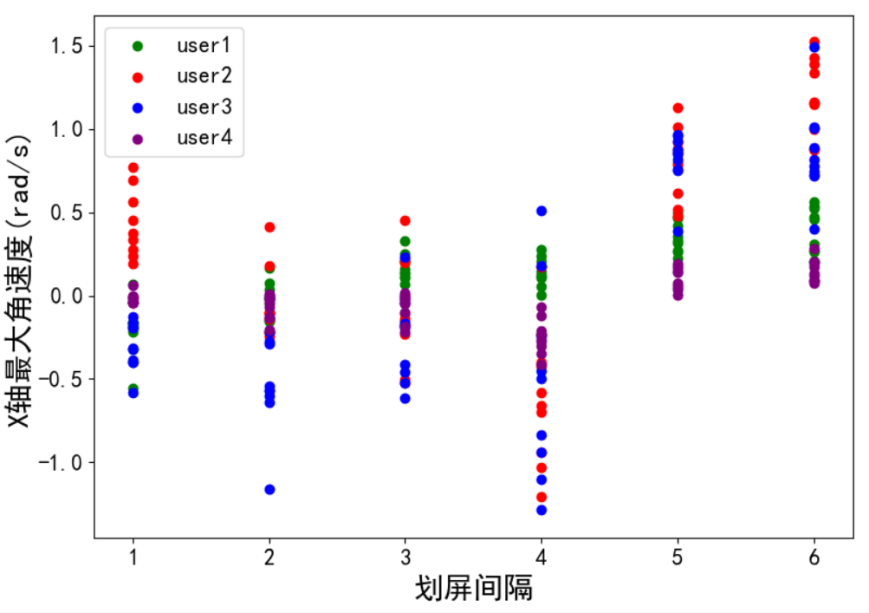


图 22 4位志愿用户触屏时x轴最大角速度

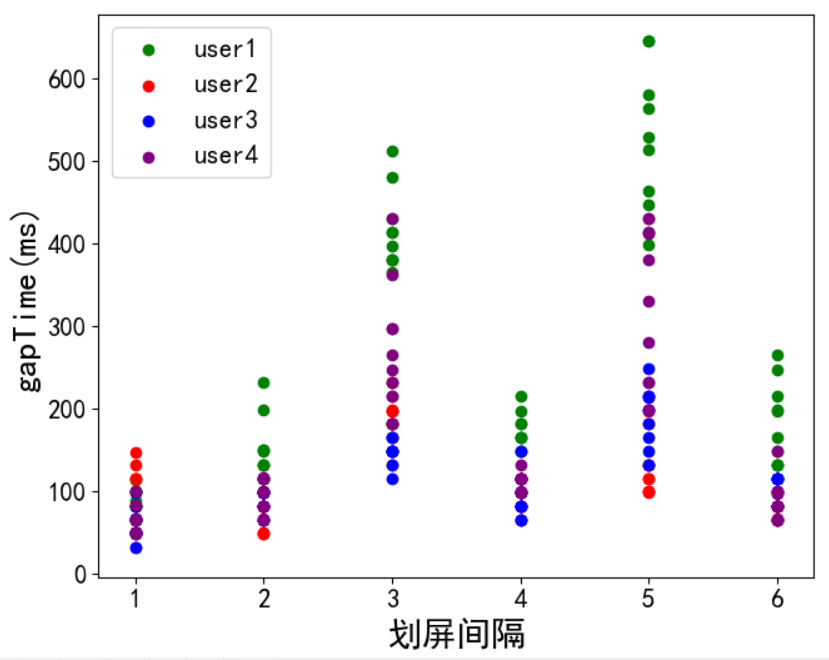
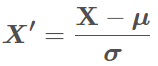


图 23 4位志愿用户触屏gapTime

1. **进行训练**

在提取预处理数据的数据之后，MVSA系统会提取出上文中所总结的传感器特征，训练用户模型。如果是二分类模型，我们在训练集中将该用户的数据作为正样本，再从数据库中随机抽取一些用户，取其数据作为负样本，二者拼成训练集。如果是无监督异常侦测，我们的训练集中只有该用户的正样本数据。为了取得更好的效果，MVSA系统对训练集进行了Z-标准化：



其中的μ和σ是先在正样本上利用sklearn的StandardScaler进行fit操作，得到正样本的均值mean\_和标准差np.srqt(var\_)，然后再使用到训练集上进行标准化操作。进行标准化之后，真正的用户和非真正用户的数据可分性更强。

之后，使用训练集训练分类器，并保存模型和计算的训练集上的参数，训练集上需要保存的参数是用户正样本的均值、方差。对于无监督异常检测模型，需要保存验证集上的MSE。

#### （2）训练模块算法实现

如果是九宫格密码，需要在该部分先计算速度特征，在数据采集时，只获取了坐标和时间戳。假设在一段时间内，用户划屏是匀速直线运动，划屏速度用位移除以时间计算得到。

|  |
| --- |
| //速度特征的计算  if data.TimeStamp[j+1] != data.TimeStamp[j]:  time = data.TimeStamp[j+1] - data.TimeStamp[j]  x\_offset = data.Xord[j+1] - data.Xord[j]  y\_offset = data.Yord[j+1] - data.Yord[j]  s = x\_offset\*x\_offset + y\_offset\*y\_offset  s = math.sqrt(s)  v = s/time |

采集到的用户数据是正样本，先计算它们的均值和方差，保存起来，为后面训练和测试做准备。

|  |
| --- |
| #先在真用户数据上进行fit操作，获取均值和方差  trueuser\_data=np.zeros((len(trueuser\_for\_train\_index),df\_feature.shape[1]))  trueuser\_data=df\_feature.iloc[trueuser\_for\_train\_index,:]  scaler.fit(trueuser\_data.values)  return (scaler.mean\_,scaler.var\_)  #用真用户的均值和方差对训练集进行标准化  X\_train=(X\_train-mean)/np.sqrt(var) |

1. **支持向量机**

训练阶段，需要加载之前预先计算出的训练集的正样本的均值和方差，训练是寻找支持向量和最佳分界面的过程，训练结束之后，保存单个用户的模型。

|  |
| --- |
| mean=np.load(meanpath+str(person)+'/'+str(person)+'.npy') #加载均值  var=np.load(varpath+str(person)+'/'+str(person)+'.npy') #加载方差  X\_train=(X\_train-mean)/np.sqrt(var)  clf=SVC(gamma='auto',probability=True)  clf.fit(X\_train,y\_train)  mkdir(modelpath+str(person))  joblib.dump(clf, modelpath+'/'+str(person)+'/'+str(person)+'.model') #保存模型 |

1. **自动编码器**

异常检测通常分为有监督和无监督两种情形，自动编码器算法是一种常见的基于神经网络的无监督学习降维方法。在无监督的情况下，我们没有异常样本用来学习，假设异常点服从不同的分布。根据正常数据训练出来的自动编码器，能够将正常样本重建还原，但是却无法将异于正常分布的数据点较好地还原，导致还原误差较大。根据还原误差，我们能判断输入是正常数据还是异常数据。

我们利python的keras模块实现自动编码器。在按键密码中，原始数据的维度是95维，编码器和解码器分别有两层，中间的编码器共有35个节点，也就是说原始数据被降到了35维。解码器根据降维后的数据再重建原始数据，重新得到95维的输出。从输入到输出的这个过程中，自动编码器实际上也起到了降噪的作用。

经过实验，我们选择的还原误差的度量是均方误差（MSE）。训练过程中，前30次迭代模型的loss会下降比较大，之后基本上不再下降，这时就可以停止训练。保存模型和验证集上的平均绝对误差，验证集上反映的往往是实际的情形，所以我们将验证集上的平均均方误差乘一个常数作为之后异常检测的阈值。

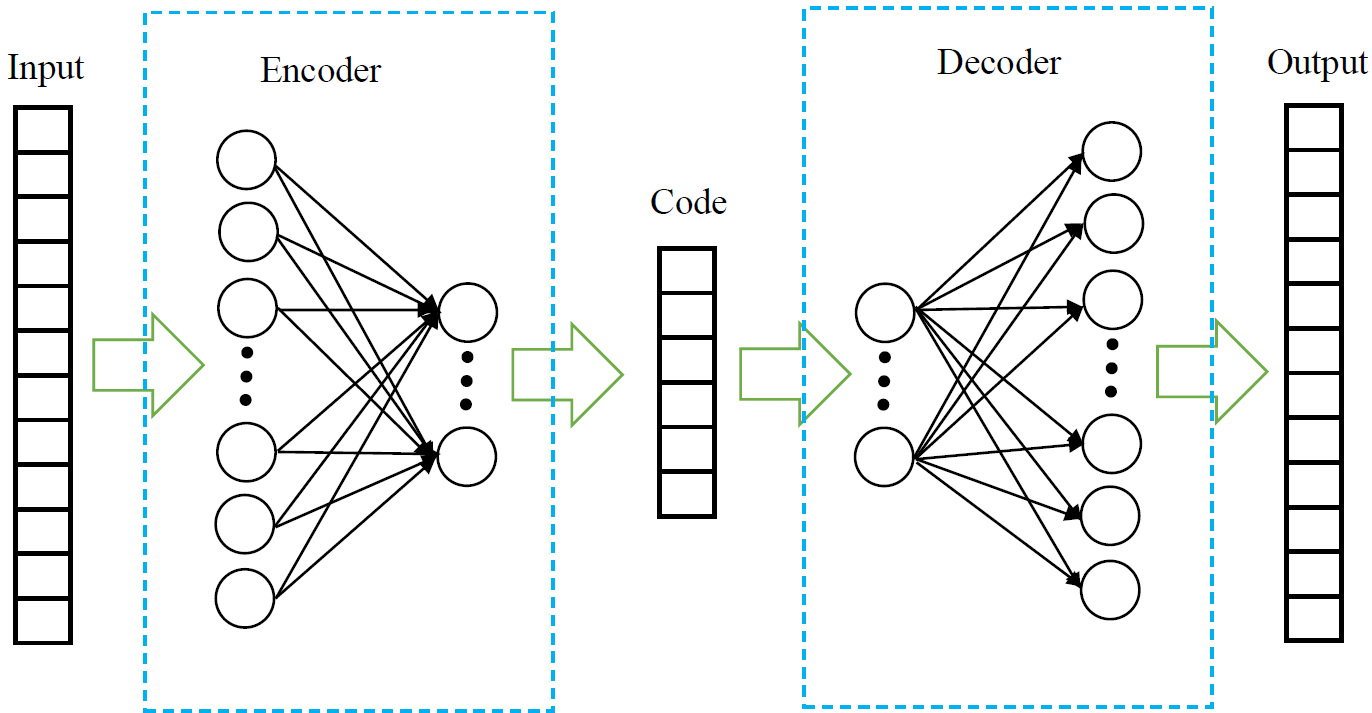


图 24 自动编码器

构建自动编码器的核心代码如下所示，这个网络共有四层，每层的激活函数分别是tanh、relu、relu、tanh，损失函数是均方差，度量是平均绝对差，在训练完成之后，保存每个用户的模型，并保存在验证集上的误差，作为之后用户登录时判定的阈值参考。对于按键密码，原始特征维度为95，我们的encoding\_dim设置为35；九宫格密码的原始特征维度为114，encoding\_dim我们设置为50。

|  |
| --- |
| input\_layer = Input(shape=(input\_dim, ))  encoder = Dense(encoding\_dim, activation="tanh",  activity\_regularizer=regularizers.l1(10e-5))(input\_layer)  encoder=Dense(int(encoding\_dim/2),activation="relu",  activity\_regularizer=regularizers.l1(10e-5))(encoder)  decoder = Dense(int(encoding\_dim / 2), activation='relu')(encoder)  decoder = Dense(input\_dim, activation='tanh')(decoder)  autoencoder = Model(inputs=input\_layer, outputs=decoder)  autoencoder.compile(optimizer='adam',  loss='mean\_squared\_error',  metrics=['mse'])  history = autoencoder.fit(X\_train, X\_train,  epochs=num\_epoch,  batch\_size=batch\_size,  shuffle=True,  validation\_data=(X\_valid, X\_valid),  verbose=0,  callbacks=[checkpointer]).history  mkdir(historypath+str(person))  #保存在验证集上的误差  np.save(historypath+str(person)+'/'+str(person)+'.history.npy',np.array(history['val\_ mean\_squared\_error '])) |

### 2.5.3用户验证登录模块

#### （1）验证登录模块算法

1. 用户在MVSA客户端输入密码；
2. 输入密码过程中采集到的数据传回MVSA服务器端，进行数据预处理，调取该用户之前存储的真实数据的均值和方差，进行数据的标准化操作。

**A.如果是二分类模型：**

1. **a** 用该用户之前训练好的模型，根据当前数据，计算信心分数；
2. **a** 若信心分数大于阈值，认为的确是真的用户输入的密码；否则，认为是冒名顶替的用户输入了密码。

**B.如果是无监督异常侦测：**

**3. b** 用该用户之前训练好的模型，根据数据，进行重建数据操作；

**4. b** 以该用户训练阶段验证集上重建误差均值作为阈值，如果该数据的重建误差小于验证集上的均值，认为是真的用户输入的密码；否则，认为是冒名顶替的用户输入了密码。

#### （2）验证登录模块算法实现

1. **二分类模型**

二分类模型验证登录的主要过程如下，先加载用户个人模型，然后进行数据预处理，读取预处理之后的数据，再加载之前真正样本的均值和方差，用加载的模型预测现在的数据来自真用户的概率，概率大于阈值，认为是真用户，否则认为是冒名顶替的用户：

|  |
| --- |
| clf= joblib.load('./model/'+str(person)+'/'+str(person)+'.model') #加载用户个人模型  t=single\_data\_transision(input\_user\_excel) #进行数据预处理  if(t==0):  print('Sorry,file wrong type, please enter again,that will be all right')  return -1  df=pd.read\_csv('wait\_judge.csv',sep=',') #读取数据预处理之后的数据  X=np.zeros((df.shape[0],len(features3)))  X[:,:]=df[features3].values  mean=np.load(meanpath+str(person)+'/'+str(person)+'.npy') #加载之前真正样本的均值  var=np.load(varpath+str(person)+'/'+str(person)+'.npy') #加载之前真正样本的方差  X=(X-mean)/np.sqrt(var) #进行标准化操作  y\_prob=model.predict\_proba(X) #用模型预测信心分数  y\_pred=(y\_prob[0,1]>judge\_thre).astype(np.int32) #信心分数大于阈值，认为是真正用户  if(y\_pred>0):  print('True user!')  else:  print('Eve!')  return y\_pred |

1. **无监督异常侦测**

|  |
| --- |
| Mae\_test = np.mean(np.abs(X - pred\_X), axis=1)  thre=np.load(historypath+str(person)+'/'+str(person)+'.history.npy')  thre=thre[int(-len(thre)\*0.4):].mean() |

自动编码器无监督异常侦测的用户验证登录过程，数据处理部分与二分类模型相似，只不过这里需要加载之前训练时存储的验证集上的平均绝对误差，取最后训练阶段的平均绝对误差的均值作为阈值。使用用户训练的模型重建数据，如果重建之后数据的平均绝对误差小于阈值，认为是真用户，否则认为是冒名顶替的用户。

当判断为真用户时，会提示认证通过，否则提示认证失败：

图 25 认证结果显示

## 2.6 本章小结

本章节主要通过对MVSA系统的功能、依托的安卓传感器设备、软件处理流程以及软件模块的描述，介绍了MVSA的具体实现的过程和运行处理流程。在后续的第三章节中将介绍对MVSA系统功能的验证测试。

# 第三章 作品测试与分析

本章主要介绍将采集到的多个用户的数据放入MVSA系统之中进行识别的测试结果。在此基础上，本章还进一步分析在测试实验之中TPR、TNR、FPR、FNR和F1-score等测试指标，以此来考量系统的准确性和可靠性。此外，对于按键密码的测试，我们还邀请了志愿者使用节拍器进行数据录入，模拟对于MVSA系统进行模仿攻击的场景，检测在模仿攻击的情况下MVSA系统的识别准确率是否会受到影响；同时，我们还录入了在不同场景下（包括但不限于，站姿，坐姿，洗澡前后，是否乘坐交通工具等多个情形）的多组数据进行测试；与此同时，本章还将我们的工作成果与以前的相关工作结果进行对比，分析。

## 3.1 测试数据集

在测试用户数据训练和用户登录模块上，我们选用采集的数据集进行测试。按键密码测试时，我们请求每位志愿者都使用右手持手机，输入相同的6位数密码“123456”。而九宫格密码测试时，我们请求志愿者左手持手机，右手划屏输入7位密码[1 2 3 6 9 7 8]。

按键密码数据库**1**中，我们收录了50位志愿者的数据，每位志愿者约录入30个有效样本，手机触屏有触摸面积数据，没有触摸压力数据。按键密码数据库**2**中，两位志愿者使用节拍器，以90拍/分钟的节奏录入样本，另有5位志愿者没有使用节拍器录入的数据，手机触屏有触摸压力数据，没有触摸面积数据。按键密码数据库**3**中，收录了1位组员在多种场景下的数据，手机触屏有触摸压力数据，没有触摸面积数据。

九宫格密码数据库中，我们收录了30位志愿者的数据，每位志愿者约录入30个有效样本，手机触屏有触摸压力数据，没有触摸面积数据。

## 3.2 测试指标

在对MVSA系统的测试中，我们采用机器学习系统的常用评价指标来衡量系统性能。主要包括TPR、TNR、FPR、FNR和F1-score。

TP（True Positive）是真阳性，即真用户的样本被判为真用户的样本数；TN（True Negative）是真阴性，即假用户的样本被判为假用户的样本数；FP（False Positive）是假阳性，即假用户的样本被判为真用户的样本数；FN（False Negative）是假阴性，即真用户的样本被判为假用户的样本数。

TPR（True Positive Rate）可以理解为所有正类中，有多少被预测成正类（正类预测正确），也叫召回率(Recall)或者敏感性（Sensitivity），TPR越接近1越好，说明真用户被正确识别的概率很高，给出定义如下：

TNR（True Negative Rate）可以理解为所有反类中，有多少被预测成反类（反类预测正确），也叫特异性（Specificity），TNR应该尽量接近1，说明假用户被正确识别为假用户的概率很高，给出定义如下：

FPR（False Positive Rate）可以理解为所有反类中，有多少被预测成正类（正类预测错误），FPR越接近0越好，说明假用户被错判为真用户的概率很低，给出定义如下：

FNR（False Negative Rate）可以理解为所有正类中，有多少被预测成反类（反类预测错误），FNR应该尽量接近0，说明真用户被错判为假用户的概率很低，给出定义如下：

F1-score（F值）又称作F1-measure，是综合考虑Precision和Recall的指标，越接近1说明模型表现越好，越接近0说明模型表现越差，给出定义如下：

MVSA系统的可靠性和可用性主要考虑如下三个指标：

（1）数据集上用户的平均TPR；

（2）数据集上用户的平均FPR；

（3）数据集上用户的平均F1-score；

## 3.3 测试方案

测试部分主要测试系统是否实现了期望的功能，以及测试系统的性能（如准确率、误报率、漏报率、检测用时）。系统的性能中，考虑到身份认证的特殊性，准确率不是关心的主要指标。在一定程度上，出现5%～10%左右的误报率是可以接受的，而漏报率应该尽量低一些，控制在10%以内。训练是在服务器端，所以用时可以比较久，但验证登录时，用时不能太久。

我们使用按键密码数据库**1**和九宫格密码的数据库，对我们的系统进行性能测试。分别对二分类模型+按键密码、无监督异常侦测+按键密码、二分类模型+九宫格密码、无监督异常侦测+九宫格密码四种组合进行了测试，求取所有用户上平均的TPR、TNR、FPR、FNR和F1-score，测试系统在数据集上的性能，验证系统的可用性和可靠性。

对于按键行为识别的数据库**2**和**3**，我们使用二分类模型进行测试，验证系统的稳定性和可靠性。

### 3.3.1 二分类模型的测试

对于按键密码和九宫格密码的二分类模型测试，训练时需要提供正样本和负样本。在数据库中，我们每次选择一个用户为真用户，数据库中的其它用户是假用户，真用户的数据是正样本，假用户的数据是负样本。训练时，真用户的数据选取70%到训练集，30%到测试集中；随机匿名抽取50%的假用户，其样本放到训练集中，另外抽取50%的假用户的样本放到测试集中。用训练集训练分类器，用测试集测试分类器，获取TPR、TNR、FPR、FNR和F1-score。对每个用户做同样的操作，并计算数据集上的平均TPR、TNR、FPR、FNR和F1-score作为模型的测试结果。

TPR反映的是真用户被正确识别的概率，TPR应该越接近1越好；FPR反映的是其它用户被识别为真用户的概率，FPR应该越接近0越好；F1-score反映的是模型的综合表现，应该越接近1越好。如果最终测试结果中TPR较高、FPR较低且F1-score较高，则说明系统工作良好；否则调整参数，调整选择的特征、代码和算法，进一步测试，直至获得较好的效果。

### 3.3.2 无监督异常侦测的测试

对于按键密码和九宫格密码的无监督异常侦测的测试，在数据库中，每次选择一个用户为真用户，数据库中的其它用户是假用户。训练时，真用户的数据选取70%，按7：3的比例切分为训练集和验证集；真用户的数据选取30%到测试集中，测试集另外的数据由所有的假用户的数据构成。用训练集训练模型，用测试集测试模型，获取TPR、TNR、FPR、FNR和F1-score。对每个用户做同样的操作，计算数据集上的平均TPR、TNR、FPR、FNR和F1-score。

TPR反映的是真用户被正确识别的概率，TPR应该越接近1越好；FPR反映的是其它用户被识别为真用户的概率，FPR应该越接近0越好；F1-score反映的是模型的综合表现，应该越接近1越好。如果最终测试结果中TPR较高、FPR较低且F1-score较高，则说明系统工作良好；否则调整参数，调整选择的特征、代码和算法，进一步测试，直至获得较好的效果。

## 3.4 测试环境

### 3.4.1硬件配置

服务器端：

Intel core i5-8300H 2.3GHz

NVIDIA Geforce GTX 1050

内存8GB

手机端：

华为mate8、小米6、vivox20，均内置基础的传感器

### 3.4.2软件配置

操作系统： windows10

python开发：python3.6.8、sklearn 0.20.2、tensorflow 1.12.0、keras 2.2.4等

Android开发：Android API 23、Android Studio、Eclipse等

## 3.5 测试过程及结果

### 3.5.1 按键密码测试

#### （1）二分类模型测试

我们先使用按键密码数据库**1**进行测试，从数据库中，每次选择一个用户为真用户，数据库中的其它用户是假用户。训练时，真用户的数据选取70%到训练集，30%到测试集中；随机抽取50%的假用户，其样本到训练集中，另外50%的假用户的样本放到测试集中。用训练集训练分类器，用测试集测试分类器，获取TPR、TNR、FPR、FNR和F1-score。对每个用户做同样的操作，计算数据集上的平均TPR、TNR、FPR、FNR和F1-score。

在测试中，我们的系统表现出了较强的可靠性和可用性，在整个数据集上，平均TPR较高，FPR较低，F1-score较高。TPR的值高于90%，正常用户有较大的概率能够正常登录，而非法用户即便知道密码，想要登录成功，概率只有0.92%。TPR并没有接近100%，一方面是由于我们的训练集中正样本比较少，每个用户训练时只有约20个正样本。另一方面是因为，出于测试和安全的考量，我们要求每个用户输入的是简单6位数密码“123456”，位数比较少，并且还不是用户平常习惯输入的支付密码。

表 3 MVSA按键密码二分类模型下测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TPR** | **TNR** | **FPR** | **FNR** | **F1-score** |
| 92.622% | 99.769% | 0.230% | 7.377% | 0.921 |

在抗模仿攻击测试数据集也就是按键密码数据库**2**中，我们有两位志愿者使用节拍器，以90拍/分钟的节奏录入样本，另有5位志愿者没有使用节拍器录入的数据。我们使用1位志愿者的数据作为正样本，随机抽取其它用户的数据作为负样本，组成训练集；另1位志愿者的数据和另一部分用户的数据放入测试集中，进行测试。

测试结果混淆矩阵如下图26及图27所示，表明，我们的系统能有效地抵抗模仿攻击，即使攻击者已知用户的密码，同时在用户输入密码时准确记下用户点击屏幕的节奏，仍然不太可能成功登录。

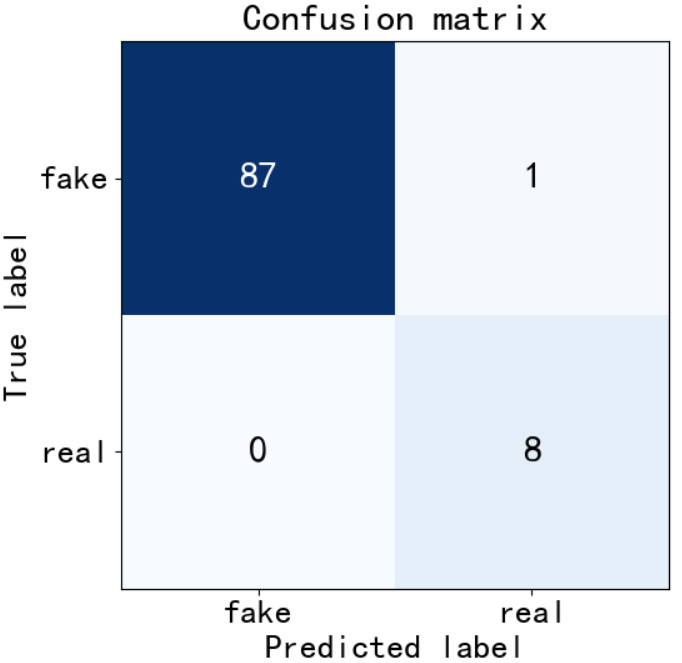


图 26 志愿者1识别结果混淆矩阵

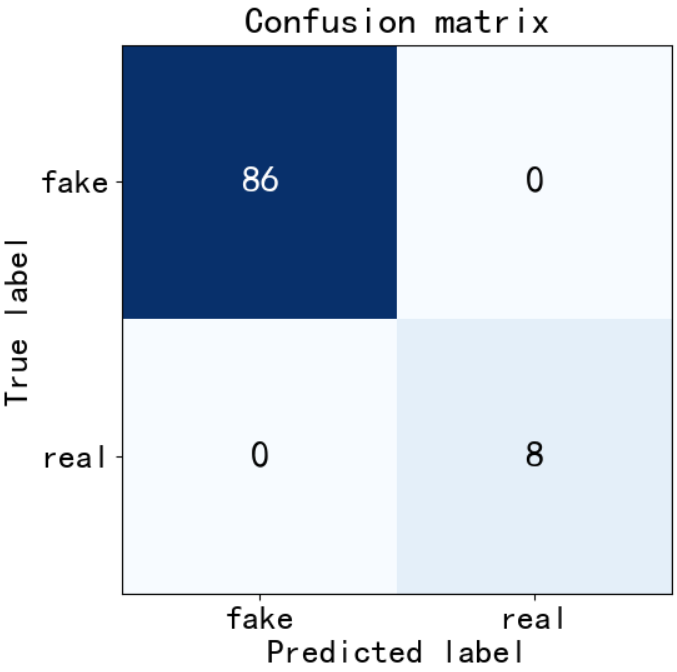


图 27 志愿者2识别结果混淆矩阵

按键密码数据库3中，收录了1位组员在多种场景下的数据，我们将其划分为训练集和测试集。我们使用数据库2中的数据为训练提供负样本，在测试时，将其作为假用户。测试结果表明，我们的系统具有较强的稳定性，该组员的TPR较高，而FPR较低，混淆矩阵结果如图28所示。结果表明，即便真用户是在公共交通工具上，或是在跑步之后，无论在站着还是坐着，使用我们的系统，有大概率能正常登录我们的系统，而其它用户被误判为该用户的可能性非常小。

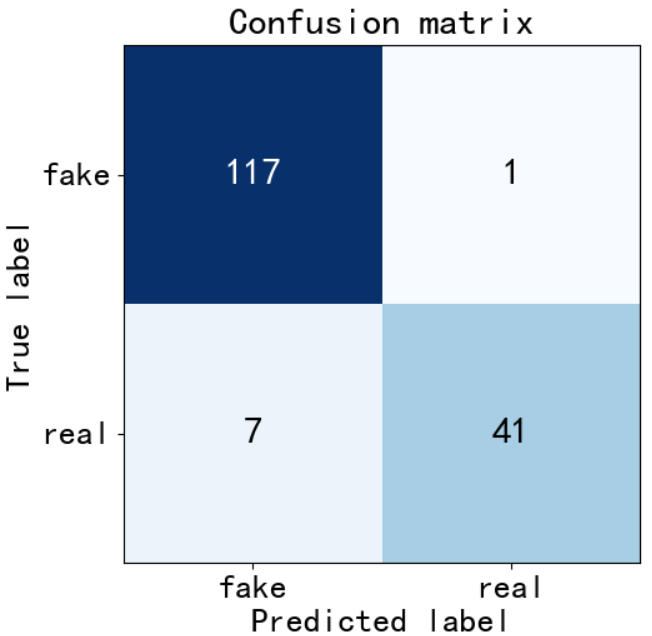


图 28 用户多场景下数据测试结果

#### （2）无监督异常侦测模型测试

我们的系统通过使用自动编码器实现无监督异常侦测模型，测试过程中，使用按键密码数据库**1**进行测试。从数据库中，每次选择一个用户为真用户，数据库中的其它用户是假用户。训练时，真用户的数据选取70%，按照7：3的比例切分为训练集和验证集；30%的真用户的数据放到测试集中，另外所有的假用户的样本放到测试集中。用训练集训练模型，用测试集模型，获取TPR、TNR、FPR、FNR和F1-score。对每个用户做同样的操作，计算数据集上的平均TPR、TNR、FPR、FNR和F1-score。

对于一个用户，在训练阶段，初期，随着迭代次数的增加，训练集上重建误差MSE会不断地显著下降，在经过几十次迭代后，验证集上的重建误差均没有明显下降，这这时就可以停止训练。如下图29所示，训练50次～80次就可以停止训练，用验证集上30次迭代以后的MSE均值作为之后判断的阈值。

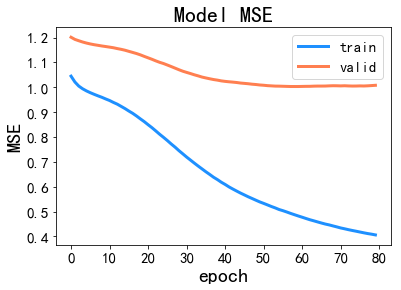


图 29 同一个用户数据迭代时的重建误差变化

而重建误差，在MAE和MSE上真用户的数据和假数据均有一定的区分度，如下图30所示，我们选择的度量是MSE。可以看到，真用户数据也就是蓝色点，重建的MSE大多在1.1以下，而假用户数据的重建误差大多在1.1以上。

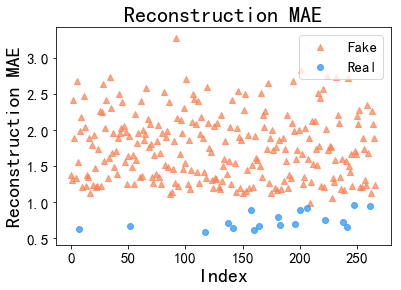


图 30 真用户的数据和假数据在MAE上的分布情况

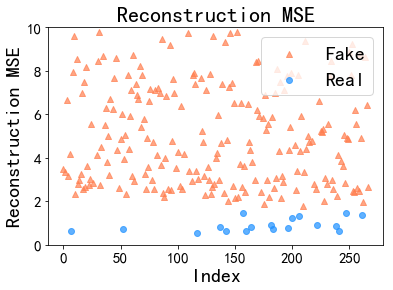


图 31 真用户的数据和假数据在MSE上的分布情况

其中某位用户的ROC曲线如图32所示：

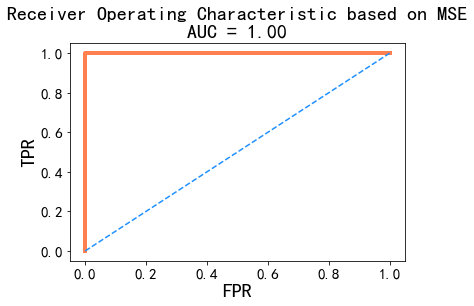


图 32 MVSA系统下某测试用户ROC曲线

在测试中，MVSA系统表现出表现出了较强的可靠性和可用性，在整个数据集上，平均TPR较高，FPR较低，F1-score较高，如表所示。无监督异常侦测模型并没有二分类模型那么稳健，这是由于我们采集的单个用户数据较少导致的。我们使用了验证集，对于总共30个样本的用户，我们的训练集大约只有14个正样本，而二分类模型中使用的是约20个正样本。在测试中，有几位志愿者录入了40～50个有效样本，他们的数据上TPR有明显的提高，接近100%，而FPR在5%以下。

表 4 MVSA系统按键密码无监督异常检测模型测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TPR** | **TNR** | **FPR** | **FNR** | **F1-score** |
| 90.660% | 93.431% | 6.568% | 9.339% | 0.555 |

在实际应用场景中，如果用户不希望自己的数据被当做其他用户训练时的负样本，希望自己的隐私数据得到保护，这时用户就可以选择MVSA系统后台使用无监督异常侦测模型。

#### （3）与现存的相关工作测试结果对比

我们的测试中，用户只输入了6位数字密码，每个人输入约30个有效样本，训练时训练集只使用了约14个样本，验证集中有约6个样本，还有10个留给测试时使用，在这种情况下，我们的结果仍然比在Margit Antal文中单分类的结果要好[13]。在Margit Antal文中，每个用户输入13位的混合字母、数字、shift键的密码51次，训练过程中使用了约45个样本，其结果如下表5所示。从比较之中我们可以看出，我们的MVSA系统用更少的训练样本，更具有私密保护性的训练模型，却达到了更好的识别效果。

表 5 MVSA与现存相关工作性能对比

|  |  |  |  |
| --- | --- | --- | --- |
|  | Classifier | FPR | FNR |
| Margit Antal | knndd | 0.13±0.10 | 0.11±0.15 |
| Margit Antal | parzendd | 0.00±0.00 | 1.00±0.00 |
| Margit Antal | mogdd | 0.02±0.03 | 0.53±0.24 |
| **MVSA** | **Auto Encoder** | **0.06** | **0.09** |

### 3.5.2 九宫格密码测试

#### （1）二分类模型测试

在九宫格密码数据库中，每次选择一个用户为真用户，数据库中的其它用户是假用户。训练时，真用户的数据选取70%到训练集，30%到测试集中；随机抽取50%的假用户，其样本到训练集中，另外50%的假用户的样本放到测试集中。用训练集训练分类器，用测试集测试分类器，获取TPR、TNR、FPR、FNR和F1-score。对每个用户做同样的操作，计算数据集上的平均TPR、TNR、FPR、FNR和F1-score。

在测试中，我们的系统表现出了较强的可靠性和可用性，测试结果如下表所示，在整个数据集上，平均TPR较高，FPR较低，F1-score较高。TPR的值约90%，正常用户有较大的概率能够正常登录，而非法用户即便知道密码，想要登录成功，概率接近0%，几乎不可能成功登录。TPR并没有接近100%，一方面是由于我们的训练集中正样本比较少，每个用户训练时只有约20个正样本。另一方面是因为，出于测试和安全的考量，我们要求每个用户划的是简单7位数密码[1 2 3 6 9 7 8]，位数比较少，并且还不是用户平常习惯划的解锁密码。

表 6 MVSA系统九宫格密码二分类模型测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TPR** | **TNR** | **FPR** | **FNR** | **F1-score** |
| 93.040% | 99.828% | 0.171% | 6.959% | 0.940 |

#### （2）无监督异常侦测模型测试

MVSA系统通过使用自动编码器实现无监督异常侦测模型，测试过程中，使用九宫格密码数据库进行测试。从数据库中，每次选择一个用户为真用户，数据库中的其它用户是假用户。训练时，真用户的数据选取70%，按照7：3的比例切分为训练集和验证集；30%的真用户的数据放到测试集中，另外所有的假用户的样本放到测试集中。用训练集训练模型，用测试集模型，获取TPR、TNR、FPR、FNR和F1-score。对每个用户做同样的操作，计算数据集上的平均TPR、TNR、FPR、FNR和F1-score。

在测试中，我们的系统表现出了较强的可靠性和可用性，在整个数据集上，平均TPR较高，FPR较低，F1-score较高，如表所示。TPR在90%以上，FPR在5%以下，F1-score高于0.5，说明即使在样本比较少、训练集中只有约14个正样本，并且九宫格划屏只划了7位数的简单密码的情况下，我们的系统依然表现出了较强的可用性和可靠性。

表 7 MVSA系统九宫格密码无监督异常检测模型测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TPR** | **TNR** | **FPR** | **FNR** | **F1-score** |
| 91.509% | 95.807% | 4.192% | 8.490% | 0.626 |

在实际应用场景中，如果用户不希望自己的数据被当做其他用户训练时的负样本，希望自己的隐私数据得到保护，这时用户就可以选择MVSA系统后台使用无监督异常侦测模型。

## 3.6 本章小结

在本章中，我们使用多个数据集，对系统进行了全方位、多角度的测试。在按键密码的测试中，系统使用二分类模型时，TPR达到91%以上，FPR在1%以下；系统使用无监督异常检测模型时，TPR达到90%以上，FPR为约6%。在九宫格密码的测试中，系统使用二分类模型时，TPR达到93%以上，FPR在1%以下；系统使用无监督异常检测模型时，TPR在91%以上，FPR为约4%。

同时，对于按键密码，我们还考虑了多场景录入的数据和模仿攻击，在这两种情景下，我们的系统的测试结果均较好，与普通情景的差异性并不显著。与按键密码的相关研究的对比，表明我们的系统性能优良。

测试结果表明，我们的系统MVSA表现良好，性能与预期基本相符，无论是在按键密码还是九宫格密码中，无论使用二分类模型还是无监督模型，都基本能满足用户的需求。

# 第四章 创新性说明

本章将从创新性与实用性两个角度来阐述本作品的特点与优势.

## 4.1 创新性

### 4.1.1 多因素认证

传统的密码验证方式如支付宝和微信的6位支付密码，只验证用户输入的密码是否正确，如果密码正确则能通过验证。传统的手机解锁密码有4位或6位密码，也有九宫格划屏解锁密码，均只对密码序列进行验证。一旦密码泄露，当用户暂离手机或手机丢失时，存在资金隐患。而我们的MVSA系统采用多因素认证机制，不仅仅局限于验证密码是否正确。

一方面，我们利用安卓平台提供的内置传感器api接口，记录下来用户输入密码时的压力序列、触屏坐标序列、触屏间隔序列、手机加速度序列、陀螺仪传感器数据序列等数据作为特征；另一方面，我们在直接获取到的传感器数据基础之上，进行了用户输入数据低阶特征的提取，例如结合触屏坐标与触屏间隔序列计算划屏速度，提取每次交互中时间序列数据的最值和平均值等。在收集、计算得到所需特征后，将数据输入系统进行训练，得到用户的认证识别模型。以往所提出的该类问题解决方法往往只顾及到某一个角度或是特征，我们的系统使用全方位的特征来描述用户输入，从而保证了较高的准确率。

### 4.1.2 九宫格划屏密码输入键盘

我们的系统MVSA提供了两种认证场景，一种是支付场景，一种是手机解锁场景，分别对应于六位数字密码输入和九宫格划屏密码输入。最初，我们设想我们的系统是服务于支付密码的，为此我们针对六位支付密码进行了设计。在取得了较好的效果之后，我们将应用场景拓展到手机的锁屏密码中，针对九宫格划屏解锁密码进行了设计。九宫格划屏密码认证效果更好，能够提取到更加全面的用户行为特征数据，例如划屏速度、偏移距离等等，全面的数据特征更有利于对每个用户进行行为特征的建模，实验结果表明该方法在九宫格划屏密码下有更高的识别准确率。

### 4.1.3 在安卓多因素认证系统这一案例中首次引入无监督异常检测模型

以往所提出的身份认证分类方法多是二分类模型，即在同时给定正样本和负样本的基础之上进行分类器模型的训练，这类分类器虽然应用效果不错，然而在实际应用场景当中，往往只有正样本，并没有负样本，即只有真用户本人的数据，除非其他用户愿意提供自己的数据作为负样本。在MVSA的实现中，我们在确保识别较高的识别正确率（93%以上）的前提之下，使用了无监督异常检测模型，更贴近现实问题的处理模式，同时也能更好地保护用户的数据隐私。

## 4.2 实用性

### 4.2.1用户操作无感

我们的MVSA系统简单易行，用户操作十分友好。MVSA与正常的密码认证操作完全相同，不需要用户提供任何额外的操作，系统隐藏了复杂的技术细节部分。用户与平常输入6位支付密码或划屏进行屏幕解锁的操作完全相同，却能够使用进行了安全增强的认证服务。

MVSA采用服务端-客户端模型搭建系统，经过实验测试验证，能准确、快速的检测出输入密码的人是否为真正的合法用户。测试结果表明，MVSA能在真实用户预先录入数据样本较少的情况下（20~30个正样本），对于未知输入数据，进行较为准确的判定（准确率93%以上）。MVSA系统能够在不影响用户体验的情况下设下安全保护网，拦截下恶意攻击者意欲利用正确密码窃取用户资金的行为。

### 4.2.2 用户使用便捷

用户初次使用MVSA系统或者重置密码时，需要预先进行数据录入。经过我们的实验，用户录入20~30个有效样本，MVSA系统就能够提供较高的认证准确率。通常情况下，用户录入这些样本只需要2~3分钟，在使用上是非常便捷的。

### 4.2.3 多场景下稳定可靠

我们在设计MVSA系统时加入了对于用户输入场景的考量。由于手机这类移动设备的便携性，用户可能在多种场景下执行系统认证操作。因此，在实验测试阶段，我们邀请了志愿者在各种不同的场景下录取实验数据进行系统测试实验（场景包括但不限于站姿，坐姿，洗澡前后，是否乘坐交通工具等多个常见认证场景）。经过测试，我们的系统MVSA在多种场景下是稳定可靠的。我们还进行了抗模仿攻击测试，测试结果表明，在多种场景下，我们的系统拥有较高的准确率，是稳定可靠的。

### 4.2.4 低资源开销

MVSA系统拥有较低的用户资源开销。我们使用的传感器均是安卓提供的低功耗传感器，对手机的电量消耗比较低。我们的数据处理、模型训练模块是由服务端完成，所以在客户端上进行的数据运算量较小，不会给用户带来较大的资源开销。

## 4.3 本章小结

本章阐述了我们所设计的MVSA系统的创新性与实用性。我们的系统旨在传统的密码认证基础上，进行安全增强，可以用于支付和屏幕解锁等场景，进一步保障用户的资金安全。我们在多因素认证系统这一案例中首次引入无监督异常检测模型，并且支持在传统的6位支付密码和在九宫格划屏解锁密码中的应用。利用用户输入的多维度特征，MVSA能够准确地检测出输入密码的人是否为真正的合法用户。在使用上，MVSA操作非常简便，只有在初次使用和重置密码时，用户才需进行数据录入，耗时约2~3分钟，之后在使用时，与现有的密码认证操作是一致的，用户操作无感。经过测试，我们的系统在多种场景下能够拥有较高的准确率，是稳定可靠的。

# 第五章 总结

## 5.1 工作总结

我们设计实现了**一种基于机器学习的安卓多因素认证系统MVSA，**在系统开发过程之中，我们逐步完成了，学习安卓应用开发、了解移动平台支付认证机制、采集用户输入数据、移动通信、数据特征提取、机器学习分类器等多项工作。

在前期的准备过程中，我们对现有的支付密码流程机制进行了调查学习，并提出了以用户输入特征为基础的多因素密码认证的初步思路。在确定思路之后，由于这一思路目前还尚未进入大众视野，我们搜集了国内外现存大量相关研究文献，在前人的提供的思路之上构建了框架，同时也发现了他们所提出方法的不足。虽然这个思路在智能手机市场尚未火爆全球之前就已经提出，但学术研究中并没有对这一思路的实用化以及商用化给出系统的方法。我们从其他的行为检测论文中得到启发，决定从用户输入时的传感器特征入手，利用机器学习已经较为成熟的数据分类器，实现认证系统。

在前期实践阶段，本小组分工明确，将命题所涉及的技术应用分配到个人，有效利用时间和精力去学习不同模块或算法的知识，并根据个人的进度调整队伍的整体计划。在收据搜集方面，我们学习掌握了安卓平台应用开发的基础以及手机传感器数据记录的api调用方法，实践了手机传感器对用户触屏数据的记录、整合。在数据预处理方面，我们学习了各类传感器数据的标准，数据含义，对它们进行的对比区分，找出相似点与不同点，解析数据，构建特征模型，并根据数据特点提取出了能描述用户使用手机触屏时的最小特征数据描述集。在数据传输方面，我们深入了解安卓平台tcp传输机制，采取了java-python交替式编程，用java编写手机客户端将记录好的数据发送给服务端，再使用python在服务端上接受数据进行数据处理以及分析在数据特征分类识别方面，我们研究了用户触屏输入时的数据特点，利用数据预处理模块对之进行关键特征信息的提取，标准化，归一化，最后利用机器学习上的多种分类器，结合实验数据实现了多因素用户输入识别的可能。

在中期的实践阶段，本小组对作品先分块各项工作进行优化测试，再整体融合得到最终作品。实践正是我们失败再不断尝试的过程。

经小组成员的合作努力，现阶段，本作品已经完成系统的整体设计与实现。设计了一个全面，准确，高效的输入特征数据采集，传输，标准化，分类的完整流程，提出了优化识别结果的数据处理的思路和方法，并综合利用无监督异常检测模型实现了一套较完整的多因素用户密码识别系统。与此同时，本小组对最终作品进行了大量的测试，各模块相互协调良好，系统运作高效有序。

## 5.2 下一步工作计划

本小组在目前的研究成果基础上，会继续针对现有不足改进：

* 微调One-Class Classifer模型结构和参数，继续训练修正用户击键特征数据的分类模型，探究更好的分类检测效果空间，致力于使其达到最好的分类数据效果
* 进一步向实用化，商用化方向做出优化，完善模块插件化调用接口的各版本各设备兼容性，完善UI提升用户体验

## 5.3 心得体会

项目开发过程中，项目组成员在指导老师的帮助下通过阅读大量的相关论文和文献自学，并针对一些疑问询问该方面的老师，掌握了许多相关技术手段，提升了自我学习和学以致用的能力。

我们学习掌握了安卓平台应用开发的基础以及手机传感器数据记录的api调用方法，深入了解了各类传感器数据的标准，数据含义，对它们进行的对比区分，同时自己撰写一个有效的tcp文件传输模块，对于手机安卓开发，硬件接口调用，文件传输通信有了更深入的理解；对机器学习各类分类器的调用使用也提高了我们的代码调试能力，对现存机器学习技术有了进一步的体会。同时，对于一个完整的特征数据集进行有效最小化模型的提出与思考使得我们对宏观数据建模处理等方面有了更加实际的理解。此外，我们对于使用机器学习进行数据特征分析也有些思考，如何在大规模的商业运营模式上部署合理而有效的特征预处理机制也是一大难点，想要真正在一个比较无人问津的蛮荒领域创造模型需要长时间的思考才能找到最为行之有效的解决办法，还要经过一系列的验证比对与权衡才能得出最优解。

我们不仅体验了完整的项目开发流程，还感受到了理论知识与应用之间的差距，学以致用才能真正掌握一门技术。

# 参考文献

[1]Emarketer. 2 billion consumers worldwide to get smart (phones) by 2016[J]. 2014.

[2]央行报告——全国76.9%成年人使用电子支付：http://money.people.com.cn/n1/2018/0814/c42877-30227885.html

[3]智能手机用户使用报告： https://www.huawei.com/cn/industry-insights/outlook/reports/HW\_U\_163773

[4]调查报告——中国电子支付市场规模近3万亿美元：https://www.linuxidc.com/Linux/2017-04/143092.htm

[5]2018年中国互联网网民数量及网民画像分析：https://www.chyxx.com/industry/201805/645412.html

[6]中国网络支付安全状况报告：http://workspace.unpan.org/sites/internet/documents/B2CN12%20中国网络支付安全状况报告.pdf

[7]Raghunathan A, Ravi S, Hattangady S, et al, Securing mobile appliances: new challenges for the system designer[C]. 2003 Design, Automation and Test in Europe Conference and Exhibition. IEEE, 2003: 176-181.

[8]Crawford H, Renaud K, Storer T, A framework for continuous, transparent mobile device authentication[J]. Computers & Security, 2013, 39: 127-136.

[8]Khan H, Atwater A, Hengartner U. Itus: an implicit authentication framework for android[C]. Proceedings of the 20th annual international conference on Mobile computing and networking. ACM, 2014: 507-518.

[10]Frank M, Biedert R, Ma E, et al, Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication[J]. IEEE transactions on information forensics and security, 2012, 8(1): 136-148.

[11]Harbach M, Von Zezschwitz E, Fichtner A, et al, It’sa hard lock life: A field study of smartphone (un) locking behavior and risk perception[C]. 10th Symposium On Usable Privacy and Security ({SOUPS} 2014). 2014: 213-230.

[12]Sen S, Muralidharan K, Putting ‘pressure’on mobile authentication[C]. 2014 Seventh International Conference on Mobile Computing and Ubiquitous Networking (ICMU). IEEE, 2014: 56-61.

[13] Antal M , László Zsolt Szabó, An Evaluation of One-Class and Two-Class Classification Algorithms for Keystroke Dynamics Authentication on Mobile Devices[C]. International Conference on Control Systems & Computer Science. IEEE, 2015.

[14]Ross A, Nandakumar K, Jain A K, Introduction to multibiometrics[M]. Handbook of biometrics. Springer, Boston, MA, 2008: 271-292.

[15]De Luca A, Hang A, Von Zezschwitz E, et al, I feel like i'm taking selfies all day!: Towards understanding biometric authentication on smartphones[C]. Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. ACM, 2015: 1411-1414.

[16]Shen C, Zhang Y, Cai Z, et al, Touch-interaction behavior for continuous user authentication on smartphones[C]. 2015 International Conference on Biometrics (ICB). IEEE, 2015: 157-162.

[17]Campisi P, Maiorana E, Bosco M L, et al, User authentication using keystroke dynamics for cellular phones[J]. IET Signal Processing, 2009, 3(4): 333-341.

[18] H. Saevanee and P. Bhattarakosol, "Authenticating User Using Keystroke Dynamics and Finger Pressure," 2009 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, 2009, pp. 1-2.

[19] Uno Andre Johansen, "Keystroke Dynamics on a Device with Touch Screen", June 2012, [online] Available: http://hdl.handle.net/11250/143992.

[20]Kang P, Cho S, Keystroke dynamics-based user authentication using long and free text strings from various input devices[J]. Information Sciences, 2015, 308: 72-93.

[21] S. Sen and K. Muralidharan, "Putting ‘pressure’ on mobile authentication," 2014 Seventh International Conference on Mobile Computing and Ubiquitous Networking (ICMU), Singapore, 2014, pp. 56-61

[22] A. De Luca, A. Hang, F. Brudy, C. Lindner, H. Hussmann Touch me once and i know it’s you!: Implicit authentication based on touch screen patterns, CHI’12 Proceedings of the 2012 ACM annual conference on human factors in computing systems, ACM, New york (2012)

[23] Orcan Alpar, Intelligent biometric pattern password authentication systems for touchscreens, Expert Systems with Applications, Volume 42, Issues 17–18, 2015, Pages 6286-6294,

[24] Sensors: https://developer.android.com/guide/topics/sensors

[25] sklearn: https://scikit-learn.org/stable/supervised\_learning.html#supervised-learning

[26] Building Autoencoders in Keras : https://blog.keras.io/building-autoencoders-in-keras.html