Touch Biometrics for Unobtrusive Continuous Authentication on Smartphones

by

Christina J. Kroeze

Submitted in partial fulfilment of the requirements for the degree
Master of Science (Computer Science)
in the Faculty of Engineering, Built Environment and Information Technology
University of Pretoria, Pretoria

Wednesday 21st January, 2015

Publication data:

C. J. Kroeze. "Touch Biometrics for Unobtrusive Continuous Authentication on Smartphones". Master's dissertation, University of Pretoria, Department of Computer Science, Pretoria, South Africa, 2015.

Touch Biometrics for Unobtrusive Continuous Authentication on Smartphones

by

Christina J. Kroeze
E-mail: christienkroeze@gmail.com

Abstract

Mobile devices such as smartphones have until now been protected by traditional authentication methods, including passwords or PINs. These authentication mechanisms are difficult to remember and often disabled, leaving the device vulnerable if stolen. This dissertation presents an unobtrusive, continuous authentication system based on biometric data collected from a touchscreen. A model is presented for implementation of such a touch biometric system on a smartphone. This model is evaluated by conducting an experiment simulating real-world touch interaction. Touch data was collected from 30 participants during normal phone use. This data was analysed using two classification algorithms to determine the accuracy of matching touch data to users. The results show that touch data is distinct between users and could be used in authentication without disrupting normal touch interaction. Furthermore, touch data could be used in its raw form without significant pre-processing.

Supervisor : Dr. Katherine M. Malan
Department : Department of Computer Science
Degree : Master of Science

"Let us think the unthinkable, let us do the undoable, let us prepare to grapple with the ineffable itself, and see if we may not eff it after all."

Douglas Adams

Acknowledgements

I would like to thank the following people who helped me through my studies:

- Dr. Katherine Malan, for taking over supervision of my dissertation. Without her detailed guidance and feedback, this dissertation would not have been completed;
- My parents and my brother for the immeasurable amount of support and love they have given me, their unwavering belief in my success and their strange eagerness to proofread my dissertation;
- My partner Mynhardt for letting me rant and for not letting me give up, for his help setting up my experiment and processing the results, and for countless cups of late night coffee;
- All my colleagues from SAP Research Pretoria and friends at the University of Pretoria, especially Masters' club, for the advice over great food and the adventures to maintain our sanity;

The financial assistance of the National Research Foundation (NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at are those of the author and are not necessarily to be attributed to the NRF.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The memory and storage capabilities of today's smartphones are comparable to those of PCs a few years back. In fact, because of their portable nature, people tend to use them to run most facets of their lives. It is also this combination of high value and portability that makes smartphones a target for thieves. However, current authentication technology requires that users lock their devices and remember more personal identification numbers (PINs), passwords or patterns. Users often do not rate security as a high priority [69], and may disable authentication mechanisms, leaving their devices vulnerable to imposters.

There is a need for a better approach to user authentication, especially on smartphones, to lessen the burden on the user to authenticate. Biometrics rely on unique human traits to identify and verify users. These are more usable since users do not need to remember their passwords or security tokens to authenticate.

Because of the popularity of the touchscreen as the main input device on most smartphones, it could be used to collect touch data from smartphone users. This data could be used to authenticate users on smartphones.

In particular, touch biometrics can be used continuously to verify whether the normal user is still using the phone and lock the phone when a user's behaviour has changed. This dissertation investigates the possibility of using the touchscreen to collect touch biometrics on a smartphone and to use this touch data to authenticate users.

In this dissertation, an investigation into previous findings serves as the basis for the development of a theoretical model of how touch biometrics can be implemented on smartphones. This model is tested using an experiment, in which touch data from participants is collected and analysed. The results show that touch biometric data is distinct between users and that classification can be done based only on touch biometric data.

The next section motivates the need for a better authentication mechanism on smartphones.

1.1 Motivation

Modern computers have evolved from specialised, bulky systems to popular and highly usable versions of their former selves. A natural progression of this computer revolution is the development of sophisticated *mobile devices*. They are also referred to as *handheld devices* or *handheld computers*. Improvements in hardware and software have enabled mobile devices to have most of the functionality that a computer would have, but in a smaller and portable device [91]. Examples of mobile devices include personal digital assistants (PDAs), tablets, netbooks and mobile phones.

This dissertation focuses in particular on *smartphones*. A smartphone is a mobile phone that includes more features than just making and receiving calls, for instance browsing the internet, taking photos, GPS navigation or playing music [61]. A smartphone is a combination of a mobile phone and a PDA. IBM's *Simon*, which was developed during the 1990s, is considered the first smartphone. *Simon* was the first device that combined the features of a cellphone and PDA, though it expressed a general trend of creating cellphones with more capabilities [39, 52]. However, the early smartphones did not really offer the usability of modern smartphones, and consumers usually carried a PDA anyway [49]. Modern smartphones have come a long way – they are sleek, thin and usable advanced technologies. Modern smartphones usually run on the *Android*, *iOS* or *BlackBerry* operating systems (OSs), each providing its own application programming interface (API) to developers, enabling them to create new applications for the devices. The main difference between a *smartphone* and *feature phone* is this API which is provided to independent developers to create custom applications for smartphones.

Smartphones may contain private information that should be protected from imposters. The device may contain sensitive personal information like emails, browser history, GPS history, saved wireless networks or internet banking login details. Smartphones face the same security concerns as any other online computer device. They are increasingly at risk of online attacks such as malware infections, exploits and privilege escalations as the use of these devices becomes more widespread. It is therefore as important to protect a smartphone from online attacks as it is to protect a traditional computer.

However, a more pressing problem relates to physical security. On one hand, smartphones are usually expensive. On the other hand they are also

very light and portable. This makes them the perfect target for thieves. Therefore, smartphones are at an even greater risk of theft than laptops or desktop computers, even though these larger devices could be more expensive. It is easier to steal a smartphone than it is to steal a laptop.

Smartphones should therefore be more secure than their stationary counterparts in terms of authentication. If a thief steals a smartphone, the risk is that the user's personal information could be used by the thief. For example, a thief may use the user's banking details to withdraw money from the user's bank account. However, many smartphones are not protected by authentication mechanisms since many users do not enable any kind of authentication on their smartphone.

The next section discusses the authentication mechanisms on smartphones briefly.

1.1.1 Smartphone Authentication

Traditionally, mobile phones were locked by entering a PIN or a password. The PIN was usually entered only when the device was turned on [34]. In this case, it was unlikely that the device would have been turned off when it was stolen, leaving the PIN essentially useless. One third of respondents to a survey in 2005 did not use the PIN at all [16].

Newer devices can generally be locked very easily, without needing to turn the device off. This can be done manually, by pressing the lock button, or automatically, when the phone has been left idle for long enough. The user then unlocks the screen by entering a PIN or password or by using some alternative authentication mechanism, such as is discussed below. However, the authentication mechanism is usually disabled by default [60, 73].

The Android OS offers a pattern lock mechanism [72] that is easier to enter than a PIN, but is also vulnerable to shoulder surfing and could be inferred by finger smudges left on the screen [67]. Furthermore, a pattern lock is easier to guess using brute force attacks because there are fewer possible pattern combinations available than when using a PIN or password. Recent research has shown that PINs and patterns could also be guessed using accelerometer readings from the smartphone while a user is entering the PIN or pattern [9]. The pattern locks were significantly easier to guess than the PINs using the accelerometer.

A picture password system has been implemented as part of Windows 8 [78]. Using this system, users draw gestures on specific memorised points of their personal photos to unlock the screen. However, picture passwords have been shown to be vulnerable to attackers who guess at likely pictures or picture areas that users may choose [15].

On Android version 4.0 a facial recognition login tool was introduced. While it is a very user friendly way of authenticating, it is not very secure and users are warned that the facial recognition tool is less secure than a PIN, password or a pattern. In fact, it can be fooled with a static image of the phone's user displayed on another phone [77].

One sensor that has not been commercially used as an authentication mechanism is the touchscreen. Touchscreens are very popular on modern smartphones, so they present an opportunity to implement an authentication mechanism that is universal across most smartphones.

1.1.2 Touchscreens and Other Sensors

Today's smartphones usually include a touchscreen. Touchscreens are seen as extremely *usable* devices, because they remove the barrier between the interface and the user that is often introduced by other pointing devices such as a mouse [76]. Touchscreens enable users to touch the interface directly, unlike when using a mouse, keyboard or other controller. This is especially true in modern smartphones, since touchscreens now have a very high sensitivity and applications have become more responsive.

Most smartphone touchscreens are *capacitive*. That is, they use the conductive nature of human skin to record touch interaction. The conductor disrupts a uniform electrostatic field around the area where the screen is touched. Capacitive touchscreens have better sensitivity than *resistive* touchscreens, which usually rely on using a stylus or fingernail and high pressure. Capacitive touchscreens are more informative about their users, since the users can interact directly with them and the need for a stylus is eliminated [48].

Modern touchscreens are also capable of receiving input from multiple fingers, leading to multi-touch gestures such as pinching the screen to zoom in and out.

Touchscreens (and other sensors) on smartphones have been considered as possible ways to measure *biometrics*. Biometrics are unique characteristics that can be used to identify people, such as fingerprints or iris patterns. Biometrics remove the burden of authenticating from the user, since the user is not required to remember a PIN or password. Biometrics represent a better way of authenticating for smartphones, and have been applied in various forms in the past, such as facial recognition or fingerprint scanning.

1.1.3 Biometrics Using Smartphone Sensors

It is conceivable that readings from a smartphone's sensors could uniquely identify users. For instance, data from the accelerometer may be used to infer information about a user's *gait* or manner of walking. Gait recognition using expensive and specialised equipment has been studied for some time and recent studies have attempted to achieve the same results using a smartphone's less advanced sensors [23].

Similar work has been done on desktop computers, detecting user behaviour with input devices. The field of keystroke biometrics, for example, focuses on recognising individuals based on their unique typing behaviour. This field has been explored for over 25 years [62]. These findings have been applied to the same kind of sensor data from a *soft keyboard* – a keyboard that is displayed on a touchscreen [68, 90].

Advanced sensors are not a requirement for biometrics on smartphones. Rudimentary call data, such as the time at which calls were placed and from which cellular tower the connections were made, were used to determine the identity of 95% of individuals in a study of one and a half million users' network data [55].

The field of touch biometrics has grown over recent years. The process of hardening passwords involves measuring the biometric characteristics of the way a user enters a password and adding this information to the password itself for authentication. Research has been done into hardening traditional passwords using touch data [8, 22, 67, 68, 90]. Recent work has also focused on using only touch data for authentication [29, 32, 50] without the need for a password.

The touchscreen is the primary input method on a smartphone and can therefore be used to record a global set of features that all smartphone users have in common. Considering the inherent usability of a touchscreen, it may result in expressive and unique usage patterns for each user.

Furthermore, because users interact with touchscreens constantly on smartphones, this data could be used for *continuous* authentication. That is, the authentication does not take place at specific times such as when unlocking the screen, but rather keeps authenticating users while they are using the smartphone. This kind of authentication could also be *unobtrusive* – users do not need to perform explicit actions to authenticate. They are constantly being authenticated as they use the smartphone normally. This could alleviate the issue of users not enabling authentication mechanisms on their smartphones.

Therefore, this dissertation's focus will be on the use of the touchscreen to continuously authenticate smartphone users for authentication.

1.1.4 Conclusion

Smartphone security can be improved by investigating the problems of users leaving devices unlocked, the use of obsolete PINs and passwords, easy to guess picture passwords or patterns and unreliable facial recognition technology. Authentication on smartphones is often obtrusive. The user needs to actively enter a password, pattern or PIN code to use the phone. If authentication can be done continuously successfully, authenticating will be less frustrating to users. The security of smartphones will also improve because users will be less likely to disable the authentication on their smartphones.

1.2 Objectives

This dissertation will address the problems highlighted in this chapter by answering the following *research question*:

"How can the security and usability of authentication be improved by using continuous touch biometrics measured by the touchscreen on a smartphone?"

The above research question will be answered through the following two *research subquestions*:

- What constitutes a good continuous touch biometric system?
- Can such a system be practically implemented?

Based on these questions, the objectives of this study are to:

- Conduct an in-depth survey of current research into touch biometrics on smartphones and other mobile devices.
- Propose a model for a touch biometric system.
- Analyse the viability of the model by setting up and conducting a representative experiment.
- Determine the importance of different touch biometric features.
- Evaluate the accuracy of the classification of touch biometric data.

1.3 Method

This dissertation proposes a touch biometric system that could be used on different versions of the Android OS and different smartphone models. To investigate whether this system could be implemented and be accurate, an explorative experiment was conducted. This experiment was conducted to examine possible ways of implementing the proposed system and how the touch data could be collected. Furthermore, the touch data collected from experiment participants is analysed to determine the overall accuracy of a continuous touch biometric system.

To conduct the experiment, participants were asked to perform a set of tasks on a smartphone while their touch data was recorded. Their names were anonymised as each participant was assigned an anonymous user ID. Thirty participants took part in the experiment.

1.3.1 Data Collection Tool

The Android OS was chosen as a platform for the development of a data collection tool as it is open source and customizable. Android provides a detailed software development kit (SDK) to developers [6], specifying the exact nature of each touch feature that can be collected using the SDK. It is therefore a reliable way to collect data. However, some versions of Android have different ways of implementing the measurement of touch data. This measurement remains standard only across similar devices running the same version of Android. In the experiment design, all participants used the same device, eliminating the possibility of two devices measuring touch data differently.

To implement the tool, the OS was recompiled and redeployed to the smartphone to circumvent security restrictions placed on third party applications trying to eavesdrop on touch events. The implementation of the data collection tool on the smartphone is discussed in Section 4.1.

1.3.2 Data

The collected data was in a raw format, containing information about each touch event captured by the phone. This was the first dataset. The second dataset was derived from the raw data and is referred to as the gesture dataset. This dataset groups touch events into individual gestures.

These two datasets were each also subdivided into two additional types of dataset. The first type of dataset was used to simulate *identification*. This dataset contained all data patterns from all the experiment participants and

is analysed to determine whether the system could identify users from their touch data. The investigation of the possible real-world accuracy of a touch biometric system required datasets that did not contain any touch data samples from the "attacker". Therefore, the *verification* datasets were created to test whether the system could distinguish between genuine users and imposters, without the system being trained on touch data from imposters.

The different datasets generated are defined in Chapter 4.

1.3.3 Analysis

The datasets were analysed according to the *informativeness* of each recorded feature, that is, the worth of each feature in terms of how much information is gained from it for the identification and verification tasks. The calculation of information gain is described in Chapter 5.1.

To determine the accuracy of the data, each dataset was classified using two classification algorithms, defined in Chapter 5. These algorithms were used to provide an indication of how accurate classification of touch biometric data could be. The classification algorithms' accuracy could be used to define what kind of errors the system would be expected to make and how often these errors would occur. A discussion of accuracy in terms of error rates and receiver operator characteristic (ROC) curves is given in Chapter 2.3.

The results of the analysis of the touch data is given in Chapter 6.

1.3.4 Limitations of the Experiment

For the purposes of this study, the touch biometric system was tested on only one smartphone to limit the differences in measurements that could arise if more than one smartphone is used.

Furthermore, participants only used the phone in a 20 minute session. Therefore, the effect of using the system for longer could not be determined. Since the system logs needed to be monitored and collected remotely, it was not possible within this study to have the participants take the phone with them for extended analysis.

Only two classification algorithms were used to analyse the data and these were kept relatively simple. The focus of the study was not to determine which classification algorithms performed the best or how to optimise them, but rather to see whether touch data could be classified at all.

1.3.5 Ethical Concerns

If touch biometrics are used as authentication in the future, the touch data of a participant could be used to gain access to their user account. Therefore, the touch data was anonymised since each user was assigned a unique ID that was not linked to their name.

Participants' responses to a questionnaire could be used to identify them. Therefore, the questionnaire was kept to non-identifying questions such as whether the participant has used a smartphone before or what level of education the participant has. The values in these questions were not published and were only used as a reference tool to analyse the data.

Clearance to perform the experiment discussed in this section was obtained from the ethics committee of the Engineering, Built Environment and Information Technology faculty at the University of Pretoria on the 4^{th} of April, 2013. The letter of ethical approval is provided in Appendix A.

1.4 Dissertation Outline

This section provides an overview of the outline of the dissertation. Chapter 2 gives background information on biometrics and smartphones. Chapters 3, 4, 5 and 6 deal with the novel contributions of this dissertation, including the proposal of a biometric system and the design of an experiment to evaluate its viability. Chapter 7 concludes this dissertation.

Chapter 2 introduces the concept of authentication using tokens, passwords and biometrics and discusses the security and usability of each of these methods. Common accuracy measurements used in the field of biometrics are described. These are the accuracy measurements used to discuss results in this dissertation. Finally, research related to biometrics on smartphones is reviewed.

Chapter 3 describes the suggested solution by providing a detailed definition of the proposed biometric system.

Chapter 4 outlines the process followed to collect data for the experiment, including extracting features and creating the datasets.

Chapter 5 describes classification algorithms that are used in the experiment and the steps used to prepare these classification algorithms for the experiment.

Chapter 6 describes the results of the experiment and provides specific details of the results related to the system's accuracy and other observations.

Chapter 7 concludes this dissertation, summarises its findings and proposes topics for future work.

Chapter 2

Background

This chapter discusses previous work related to this dissertation. Section 2.1 introduces the conflicts between usability and security. Section 2.2 introduces the concept of authentication and discusses the limitations of the password. Section 2.3 summarises how accuracy was assessed in this study. Section 2.4 discusses previous applications of biometrics on smartphones or other mobile devices.

2.1 Security and Usability

Authentication is usually the first place where users encounter computer security. It is also one of the most vulnerable points of a computer system, because a careless user could expose the system to an attacker. Authentication represents the point at which users are trusted to maintain the security of the system. Therefore, the security of the system becomes the responsibility of the user.

However, users are often careless about security, sacrificing security in favour of achieving their tasks faster [1, 25, 85]. They take the "path of least resistance". Security and usability are two goals that are often at odds with each other [1, 10, 14, 20, 69, 89].

Computer security aims to protect system assets from attackers and to keep certain assets from unauthorised users. It is meant to lock the system away from certain users, while still ensuring that the correct users always have the access they need. Usability relates to how easy and effective a system is to use. Often the goals of security and usability are conflicting. This section investigates the goals of computer security and usability and how they relate to each other.

2.1.1 Computer Security

A computer system's security depends on the protection of the system's assets. System assets are the hardware, software or data that form part of the system. Assets are protected in computer security by ensuring their *confidentiality*, *integrity* and *availability* [63].

Confidentiality ensures that assets remain secret and private. Restricted reading rights protect the confidentiality of assets. For example, confidentiality has been compromised if an attacker discloses secret documents such as confidential medical records.

Integrity protects assets from being modified by unauthorised users. Restricted modification rights, including the right to delete or add assets, protect the integrity of assets. For instance, if a transaction record is deleted from an accounting database by an unauthorised user, the database's integrity has been compromised.

Availability of the system's assets refers to the correct users being able to access the system's assets when required. Ensuring that a system stays online and accessible to users protects the availability of assets. Maintaining the possibility to access assets to all authenticated users ensures the availability of assets. For example, if a website becomes unavailable as a result of a denial of service attack, the website's availability has been compromised.

Hence it is as important to ensure that a system remains *accessible* to legitimate users as it is to ensure that it is kept shielded from attackers.

2.1.2 Usability

A computer system's usability relates to how easy and effective the system is to use. The goals of usability are to ensure that the system is *effective*, *efficient* and *safe* to use [76]. It also concerns ensuring that it is easy to *learn and remember* how to use the system [59] – its *learnability and memorability*.

Effectiveness is a general goal describing whether or not the system actually achieves what it is meant to. In other words, does it enable users to do what they need to do? A system could provide many features that are easy to use, but the core of usability depends on whether these features are what the users actually need. For example, accounting software is effective if it allows the user to record and process various transactions.

Efficiency relates to how long it takes users to achieve their tasks. Therefore, an efficient system would support the most common tasks using the smallest number of steps possible. For example, storing a user's payment information on an online shopping site makes the system more efficient, since users do not need to enter their payment details for every transaction.

Safety, from a usability perspective, ensures that users are not able to perform dangerous actions by accident. This involves keeping users from being able to perform certain actions, prompting users when they are about to perform dangerous actions and offering recovery options in case the user has made a mistake. For example, offering an undo button in a text editor increases the safety of the system by allowing the user to recover from a dangerous action such as deleting a block of text.

The system's *learnability* is how easy it is to learn to use the system the first time, while its *memorability* relates to how easy it is to subsequently remember how to use the system.

A system needs to achieve a good level of usability in order for it to be marketable. A system could provide the means to achieving users' tasks, but if it is not usable, the users will not be able to understand how to achieve their tasks. Therefore, if a system is not usable, it is also not useful.

2.1.3 Security vs Usability

Security experts will design a system to maintain the confidentiality, integrity and availability of its assets. Conversely, usability experts will design a system to be effective, efficient and safe to use. They will also try to increase the system's learnability and memorability. If a system is not secure, it will not be used by anyone concerned for the protection of their assets. If a system is not usable, it will not be successful, since users will not be able to achieve their goals when using the system.

If a system is completely open, with no authentication or special permissions required, it will be very usable. However, it will not be secure. Security and usability are not mutually exclusive, since both support the same task of offering a service to users. Security ensures that the system remains operational, while usability ensures that the system is used effectively.

The availability of a system is closely related to its usability since both attempt to ensure that the system is accessible to users. An unusable system also has low availability. Similarly, if a system is not available, it cannot be used. In the subsequent discussions of security and usability, security refers only to the overall strength of the protection of the system – the confidentiality and integrity of the system. This definition of "security" and "secure" will be used throughout subsequent discussions.

Thus, there exists a trade-off between a system's usability and security [1, 10, 14, 20, 69, 79, 85, 89]. It is the goal of the designers of a system to achieve a good balance between usability and security.

The next section investigates authentication more closely and evaluates various authentication mechanisms according to their security and usability.

2.2 Authentication

Authentication allows a system to distinguish genuine users from imposters. Authentication enables the system to be secure by providing the distinction between those users who have permission to access or modify certain assets and those who do not. Authentication is the first line of defence of a computer system's security – it guards the access points to the system. It is therefore an important part of a computer system's security. Authentication on smartphones is the main focus of this dissertation.

Authentication is based on three qualities: *ownership*, *knowledge* or *inherence* [42, 63, 86]. Ownership refers to something the user has, such as a security token or bank card. Knowledge is something the user knows, for instance a text-based or graphical password. Inherence is something that is unique to the user – something the user is, for example an iris pattern or a fingerprint. Some authentication mechanisms use a combination of one, two or all three qualities. For example, an ATM uses a token (the bank card) in combination with a password (the PIN). This type of authentication is called *two-factor* or *multi-factor* authentication and is most often the combination of a password with either a biometric or a token [3, 88].

Some methods of authentication are more secure or more usable than others. For example, a secure password is usually difficult to remember and not very usable. However, if users write down their passwords, it decreases the passwords' security while increasing the passwords' usability. Despite their popularity, passwords are generally considered to have low usability and low security, because they rely on the user to remember them and to make them difficult to guess [1, 11, 12, 25, 30, 87].

Biometrics, however, are easier to use because they do not place extra cognitive load on the user by requiring them to remember another password. As long as the users' biometric characteristics have remained the same, they should be able to authenticate successfully.

Both the security and usability of biometric systems depend on the accuracy of the system. The accuracy refers to the number of errors that the system makes. There are two general types of errors in biometric systems – false acceptances and false rejections [86]. Too many false acceptances lower the security of the system, by allowing imposters to log in as genuine users. Too many false rejections lower the usability of the system, by requiring genuine users to attempt to authenticate too often. These error rates are discussed in detail in Section 2.3.

Therefore, the security and usability of a biometric system is dependent on the error rates. It is possible to achieve a very secure and usable biometric system. Contrastingly, the security and usability of a password depends on

the users' behaviour. A system may be exposed if a single user set an easy to guess password or wrote down a password and an attacker gains access to the account. Rules regarding the creation of passwords can increase or decrease the security and usability of a password, but the user still needs to remember a unique password and could still choose to circumvent these rules. Biometrics have a better chance of being both usable and secure and do not rely on the user acting securely.

This section discusses authentication in more detail. Authentication is the focus of this dissertation because of its importance in maintaining the security of a system. Without authentication, the system would not be able to distinguish legitimate users from attackers. The next sections provide more details of the relative security and usability of various authentication mechanisms, starting with a brief discussion on token-based authentication in Section 2.2.1. Knowledge based authentication (passwords) is discussed in Section 2.2.2. Inherence based authentication (biometrics) is discussed in Section 2.2.3.

2.2.1 Tokens

Tokens are ownership-based authentication mechanisms. If a user has the correct token, the user is granted access without further confirmation of their identity. Although they are commonly used alone in everyday life, such as house keys, concert tickets or gift vouchers, tokens are seldom the only authentication mechanism in high risk security systems.

Tokens are easy to use since the user does not need to remember a password. However, they become a burden to the users because the users need to carry their tokens with them whenever they are going use the system [3]. Similarly, if a token gets lost, an attacker can gain access to the system. Therefore, tokens also suffer from security concerns because of the relative ease of copying or intercepting the tokens of careless users.

Tokens are often used as part of *two-factor authentication* systems where a one-time password is sent to a small device or to the user's cellphone [3]. Two-factor authentication is included on many online services such as Google Accounts [37]. This kind of authentication relies on only the genuine user having physical access to their devices. However, there are several ways of eavesdropping on one-time passwords sent using a two-factor authentication system [70].

2.2.2 Passwords

Text-based passwords are used most often as an authentication mechanism, since they are easy and cheap to implement [12]. However, users need to authenticate on a large number of systems leading to a large number of distinct text passwords that they need to remember. While these passwords were traditionally limited to use in the workplace, users are now generating more passwords for personal use without the guidance of an expert [79]. Because of the number of passwords users are expected to remember, passwords still suffer from numerous security concerns.

For example, users reuse their passwords between different systems, thereby compromising the security of one system if the other system is attacked. In their study of the password habits of 544960 users, Florencio and Herley [30] found that an average of 5.67 different websites shared the same password per user.

Furthermore, some users set weak passwords to lessen the difficulty of remembering them. In one study, an employee suggested using his wife's maiden name as his password, believing it unlikely that an attacker could obtain his personal information and deduce her maiden name [1]. A study of British office workers revealed that 47% of their passwords were based on a family member's name [5]. Attackers often research their targets before attempting to access the system. Information such as a maiden name or family names are easy to acquire using social networks. Similarly, attackers can go *dumpster diving*, searching through their targets' trash to get sensitive information from bills and other discarded documents. Recent investigations have also shown that users still choose commonly used PINs and passwords such as "1234" [4], "password" [81] or the top row of the keyboard [36]. If users set commonly used passwords, it is easier for attackers to use a dictionary of common passwords to gain access to the system.

Some users choose to write down their passwords in order to remember them, allowing them to choose more complex passwords. It has been suggested that this behaviour could actually increase security. Security experts Jesper Johansson and Bruce Schneier have both said that users should be allowed to write down their passwords [71, 80]. However, there exists a possibility that the written down passwords could be found by an attacker.

Passwords could be randomly generated and assigned to users to reduce the problem of users choosing weak passwords. However, while generating random passwords may increase the security of the passwords, it may also reduce the usability. Randomly generated passwords were found to be the hardest passwords to remember in a study by Yan et al. [87].

Systems often use a set of rules to limit the possibility of users choosing

easy passwords [33]. These password guidelines vary between systems, but usually include the following suggestions: to use uppercase and lowercase characters and special characters and numbers, to avoid words that can be found in the dictionary and to avoid family names. The guidelines suggest that passwords should be memorable, but not easy to guess; they should be as long as possible, but should never be reused; they should contain as many special characters as possible, but still be meaningful to the user [87]. Therefore, the more memorable and usable the password is to the user, the less likely that it will fall within these guidelines. Users sometimes react by setting the weakest password that conforms to these guidelines, by making predictable changes, such as capitalising the first letter, or adding numbers and special characters to the end of the password [84]. Any set of publicly available password guidelines may decrease the security of the passwords. Attackers can easily use the guidelines to alter their strategy and guess at passwords more accurately, since they know what pattern the passwords will follow [11].

This leads to security vulnerabilities and the need for more user friendly password authentication. Single sign-on (SSO) technologies may be used to manage the authentication of one user on multiple systems. However, single sign-on (SSO) is used most often within organisations [56]. Some alternatives exist for private password management. Personal password managers may aid users when they need to use multiple passwords on a web browser, by assigning only one password to unlock others [47], but this creates a single point of failure which could expose several different systems to attacks. Account managers such as Google ID, Facebook and OpenID allow users to access multiple systems using one account [83]. However, each system using the account may have access to information that the user does not want to expose. Account managers also create a single point of failure for multiple systems.

Users may become frustrated by the endless array of passwords that they need to generate and remember. It also seems that there is not much that users can do to improve the usability of their passwords without sacrificing the security of their passwords.

It is suggested that biometrics can offer some relief from the usability problems associated with passwords. The next section introduces biometrics and discusses how they may be used to improve authentication.

2.2.3 Biometrics

Biometrics are unique physiological or behavioural characteristics that are automatically measurable and distinct between individuals [86]. These tra-

ditionally include fingerprints, voice or iris patterns and facial structure. Some biometrics are used commonly in everyday life. For example, people usually provide generally identifying characteristics such as height, weight, gender and ethnicity when participating in surveys, visiting the doctor or when identifying themselves to others without visual aids. Biometrics are often used for authentication, forensic evidence, or as an anti-fraud measure (for example detecting changes in a credit card user's behaviour) [42].

Biometric systems can be used for *identification* or *verification* [42]. Verification compares a person's biometric sample to an enrolled sample and answers the question "Is this person who they claim to be?" while identification answers the question "Who is this person?".

Verification only needs to decide whether a user's data matches a specific enrolled pattern; identification needs to pick which enrolled pattern matches a specific user's data. Therefore, verification is a one-to-one relationship that needs to be tested. Identification is a one-to-many relationship that needs to be tested.

However, if a system provides identification for authentication, rather than verification, it could increase the usability of the system. The usability is improved by not requiring users to first provide a username or other credential to use for verification. However, it may decrease the security of the system since an attacker does not need to guess the target user's username along with performing a mimicry attack on the target user's biometric data.

Using biometrics may increase the usability of authentication mechanisms, since users do not need to remember a token or a password. They are authenticated by who they are. Biometrics can also increase security since users need to be present when the authentication is done. However, as is the case with most security mechanisms, biometric systems can be fooled by a skilled attacker. For example, an attacker may make a gelatine copy of a fingerprint to try and fool a fingerprint scanning system [53].

A biometric system and the biometric characteristics it uses should conform to a set of goals and performance criteria. To discuss various biometric systems, these criteria are introduced in the next section.

Evaluating biometric characteristics and biometric systems

According to Jain et al. a biometric characteristic should be *universal*, *distinctive*, *permanent* and *collectable* [42]. While these represent a perfect biometric characteristic, most systems attempt to only achieve a good balance between these goals.

For a biometric characteristic to be *universal*, every person should have it. Practically, this means that a sufficient percentage of the population should

have it. For example, most people have fingerprints, but some may have very faint imprints in the ridges of the finger or may be an amputee. Therefore, fingerprints are generally universal, but are still not perfect in representing the entire population.

Distinctiveness refers to how much a biometric characteristic varies between people. A good biometric characteristic will have a large variance between different people. For example, fingerprints are distinct even between identical twins since the ridges are formed during the first seven months of foetal development [43]. Fingerprints are, therefore, distinct enough to distinguish between genetically similar people. In contrast, some biometric characteristics may be very similar between siblings, such as voices or facial structure. These characteristics are therefore not distinct enough and would not be different enough between genetically similar people. Practically, it should be distinct enough for the system to make a reasonably accurate decision.

Permanence refers to how little the biometric characteristic changes over time. A biometric characteristic should stay stable for long enough so the system can compare a new biometric pattern to stored patterns. For example, a person's voice may change as they age, but the voice can still remain stable over the period of one day. However, a person could become ill and their voice may change as a result. Therefore, the voice could not be used as a reliable biometric characteristic until the person has healed.

Finally, the *collectability* of a biometric characteristic refers to whether it is measurable and quantifiable. In other words, whether the biometric characteristic can be collected and whether it is possible to compare the biometric characteristic's pattern to a set of stored patterns. Some biometric characteristics are easy enough to collect and are used often. For example, fingerprints are easy to collect and can be used to quickly identify a person using a database of possible matching fingerprints. However, some biometrics are more difficult to collect and compare. For example, the process of matching DNA sequences can be very complicated and take longer than a fingerprint match. Practically, a biometric characteristic should be easy enough to collect and compare to other patterns for the intended purpose.

While these goals give a guideline to the definition of a good biometric characteristic, they do not represent the overall effectiveness of a biometric system. For a biometric system to be usable and secure, the following criteria should be considered.

A biometric system should have good overall *performance*. Performance represents the accuracy of the system in accepting and rejecting users as well as the efficiency associated with doing so. This includes the available resources to do the matching and the possibility of achieving acceptable error

rates within reasonable time. Accuracy rates for biometrics are discussed in detail in Section 2.3.

A biometric system should also be *acceptable* to its users. The users should be willing to allow the biometric system to identify them by collecting their biometric information. Users may refuse to accept biometrics if they are seen as intrusive or as an invasion of privacy.

Finally, a biometric system should be difficult to *circumvent*. Circumvention refers to the overall security of the system in terms of how easily the system can be bypassed by an attacker. Biometric systems are vulnerable to the usual attacks that may be used on authentication systems including eavesdropping, playback, denial of service (DoS) and database attacks. However, biometric systems have unique vulnerabilities to attacks at the sensor level, called mimicry or spoofing attacks [86]. This is when imposters use disguises or copies of biometric samples to fool the system into believing they are someone else. Biometric systems control this vulnerability by implementing liveness tests to ensure that a "live-and-well" human being is using the sensor [54]. However, this control may not be implemented in commercial systems, especially in older or lower cost systems where security is not a primary concern. For example, in a 2002 study by Matsumoto et al. [53] none of the 11 tested fingerprint scanning systems could distinguish between fake gelatine fingers copied from enrolled users and the enrolled users themselves.

Continuous authentication using behavioural biometrics

Biometric characteristics can be classified as one of three types, namely *genetic* characteristics, *phenotypic* characteristics or *behavioural* characteristics [86]. Genetic characteristics are inherited features determined by a person's parents, such as their facial structure. Phenotypic characteristics are developed as an embryo and are unique even between genetic twins. Behavioural characteristics are learned or trained behaviours that can change over time. Examples include users' behaviour when typing on a keyboard or how they walk (their gait).

The types of biometrics are not mutually exclusive. The way people walk could be determined by both their genes and the experiences they had during their lives, making it a combination of a physiological and a behavioural characteristic. Jain et al. [42] suggest that a simpler division is clear between all physiological traits (including genetic and phenotypic) and behavioural traits.

This dissertation will focus on behavioural biometrics since they easy to measure using existing sensors on smartphones. For example, gait dynamics have been successfully measured using the accelerometer on a smartphone

[23, 40].

Furthermore, behavioural biometrics are well suited to *continuous (or unobtrusive) authentication*. Continuous authentication is usually associated with biometrics such as keystroke dynamics, which can be measured without user intervention [21, 82]. Continuous authentication detects anomalies in user behaviour and can therefore secure a system at all times against imposters. It measures behavioural biometrics unobtrusively while the user continues work as normal. This makes the authentication mechanism more usable and more secure as well, since the user is not interrupted and the system keeps checking whether the current user is still who they are supposed to be.

However, the user's work could be interrupted frequently if the continuous authentication system rejects the user too often. Therefore, it is important to balance the threshold used to decide when users are genuine or not. This balance should not decrease the security of the system.

Furthermore, there may be a window of opportunity during which an attacker could perform a dangerous action. Therefore, the time between detecting a change in behaviour and initiating a lock-out should be considered carefully.

To ensure that a system's security is not compromised by using continuous biometrics, the system could provide a fall-back mechanism that allows the user to use a traditional authentication mechanism to gain access in case of a lock-out. In this case, the fall-back should not be used so frequently that it constitutes the same effect as having no continuous authentication.

The work in this dissertation will focus mainly on continuous behavioural biometrics as it is a way of improving both usability and security of smartphones within the limits discussed above. Specifically, this dissertation will investigate touch biometrics since touch biometrics are easily measured using the touchscreen, which is a popular sensor on most smartphones. To investigate touch biometrics, a *biometric system* is proposed.

Biometric Systems

In this dissertation a *biometric system* is developed for continuous authentication using touch biometrics. The system has four modules as discussed by Jain et al. [42], namely the *sensor*, the *feature extractor*, the *matcher* and the *database*. These modules are shown as they would be used for either verification or identification in Figure 2.1.

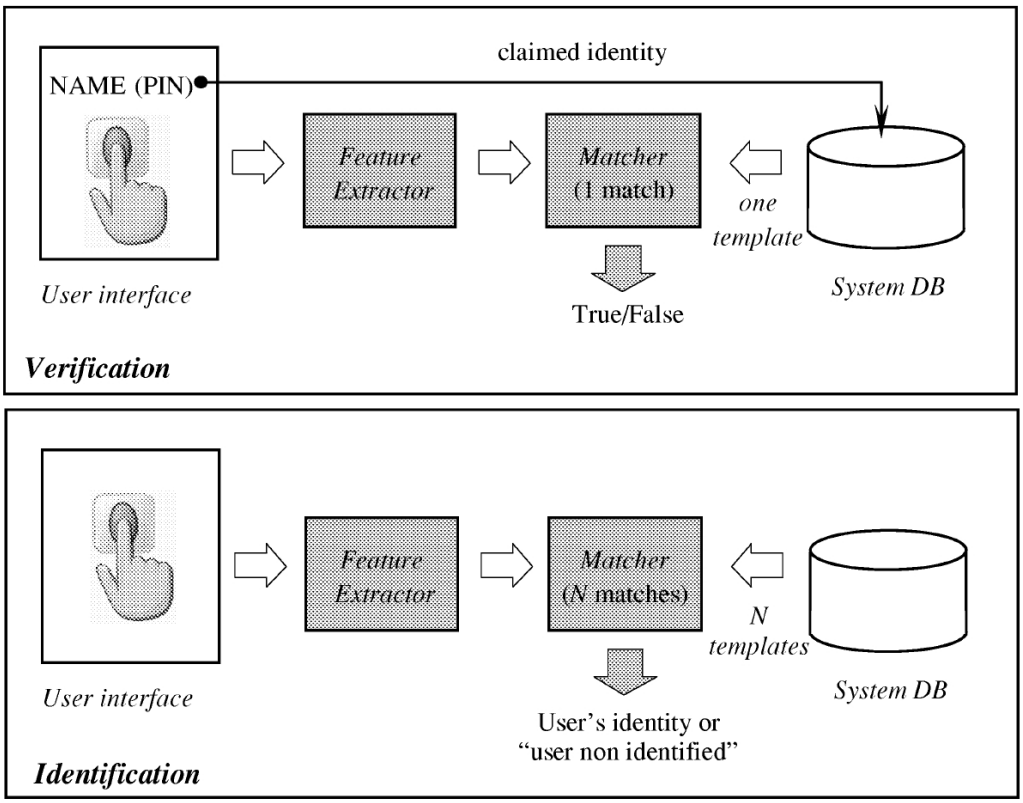


Figure 2.1: Biometric modules as defined in [42]

The *sensor* is used to collect the biometric data. For example, a fingerprint scanner is used to collect users' fingerprint data.

The *feature extractor* manages translating the collected biometric data into a set of meaningful features. In a fingerprint system, the feature extractor uses the image of the fingerprint and generates a number of points to use in comparison to other fingerprints.

The *database* is used to store the extracted features of enrolled users. This would be the central place where all the fingerprint features are kept to be compared.

The *matcher* is used to compare the biometric data presented by a user to the stored set of biometric features and determines whether the user is who they claim to be. For example, in a fingerprint system the matcher would take a newly presented finger's features and compare them to a set of stored fingerprint features.

Chapter 3 describes the proposed biometric system for touch biometrics.

To evaluate previous work related to touch biometric systems and the touch biometric system proposed in this dissertation, a set of metrics needs

to be introduced. The next section will discuss the various error rates and measurements associated with biometrics and what constitutes an *accurate* biometric system.

2.3 Measuring Biometric Accuracy

To compare the accuracy of biometric systems, various accuracy rates may be used. These accuracy rates do not only give an impression of the quality of the biometric system, but also give an indication of the purpose for which this system would be most useful. For example, if a system is very strict and has many false negatives, it would be more useful as a high security system. In contrast, if a system is less strict and has many false positives, it would be more useful as a system that identifies users to customise their online profile.

Commonly, the false accept rate (FAR), false reject rate (FRR), the overall percentage correct classifications and the values related to the receiver operator characteristic (ROC) curve are used to measure biometric accuracy [42, 86]. This dissertation will mainly use these measurements to discuss related work and empirical results, because these values give a detailed overview of the system's *accuracy* [13]. The term *accuracy* is used to refer to the overall combination of the error rates and the ROC curve.

Section 2.3.1 discusses the false accept rate (FAR), false reject rate (FRR), true positive rate (TPR), true negative rate (TNR) and the equal error rate (EER). These values are used to determine how well the system would perform as either a highly secure or a highly usable system. In other words, it determines whether the system is more conservative (has many false negatives) or more liberal (has many false positives).

Section 2.3.2 explains the receiver operator characteristic (ROC) curve and how it is applied to biometrics. The ROC curve and the area under the ROC curve are used to give a general impression of a system's performance for different levels of the threshold (strictness).

These accuracy rates will be used in subsequent discussions of previous work in the field of touch biometrics and to analyse the accuracy of the proposed touch biometric system in this study.

2.3.1 Error Rates

The false accept rate (FAR) is the probability of accepting users who are not who they claim to be. The FAR is also referred to as the false match rate (FMR) or the false positive rate (FPR). The false reject rate (FRR) is the probability of rejecting users who really are who they claim to be. The

FRR is also referred to as the false non match rate (FNMR) or the false negative rate (FNR).

The FAR and FRR are reported in terms of the number of samples from both genuine users and imposters. Therefore, if 1 out of 10 imposters were classified as genuine, the FAR is 0.1. It is not reported in terms of the total number of data instances, but rather in terms of the number of imposters.

An accurate biometric system has both low FAR and FRR rates. The exact rates that constitute an "accurate" system depend on the application of the system and the required accuracy. Generally, values should approach 0 and never be higher than 0.5. A system that has error rates that are higher than 0.5 is performing worse than randomly selecting a decision value.

Formally, the FAR and FRR are defined as:

$$FAR = \frac{p_i}{N} \tag{2.1}$$

$$FRR = \frac{n_i}{P} \tag{2.2}$$

Where:
p_i is the number of imposters that were *incorrectly* accepted,
N is the total number of imposter data instances and
n_i is the number of genuine users that were *incorrectly* rejected,
P is the total number of genuine user data instances.

The converse of the FAR and the FRR measurements, the true positive rate (TPR) and true negative rate (TNR), respectively, may also be used to compare classification accuracy. The TPR and TNR are defined as:

$$TPR = \frac{p_g}{P} \tag{2.3}$$

$$TNR = \frac{n_g}{N} \tag{2.4}$$

Where:
p_g is the number of genuine users that were *correctly* accepted,
P is the total number of genuine user data instances and
n_g is the number of imposters that were *correctly* rejected,
N is the total number of imposter data instances.

A low FAR is desirable in a high security application and a low FRR is desirable in a high usability application [42]. Therefore, a low FAR indicates that a system is more secure (fewer imposters will be accepted as genuine

users), while a low FRR indicates that a system is more usable (fewer genuine users will be rejected as imposters).

The equal error rate (EER) is the point at which the FAR and the FRR are the same. That is, the point at which the threshold for rejecting or accepting a user results in the same number of false acceptances and false rejections. This is shown in Figure 2.2.

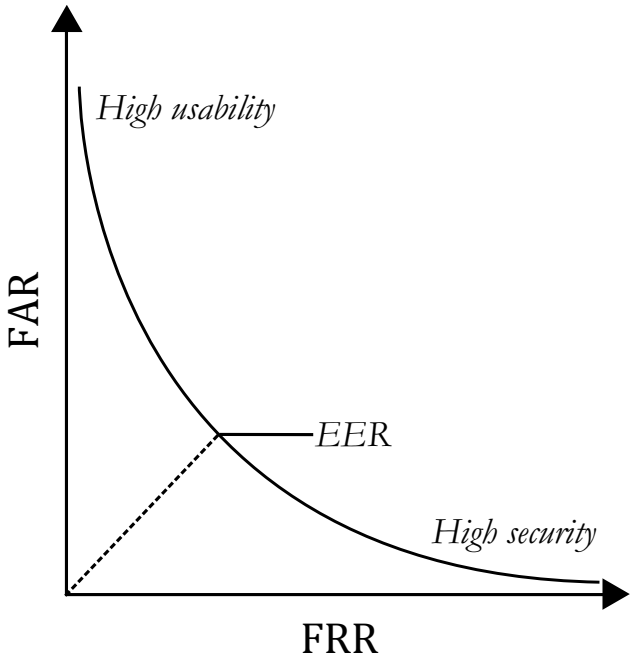


Figure 2.2: The tradeoff between high a FAR and a high FRR, where the EER is the point at which they are balanced.

A single accuracy rate or percentage correct may also be reported. This is the percentage of instances that were classified correctly. For each class, it is the sum of the percentage of true positives and true negatives in the entire dataset.

2.3.2 The receiver operator characteristic Curve

The receiver operator characteristic (ROC) curve is a plot of the TPR against the FPR for different threshold values [13] as shown in Figure 2.3. The ROC curve is a better indication of an algorithm's accuracy than a single accuracy rate, since varying threshold values can have a large effect on accuracy [28].

Any ROC curve which approximates the positive linear function is seen as worthless in terms of the accuracy of the classification, as it does not perform

better than a random selection algorithm would. Therefore, the ROC curve should always be primarily in the upper triangle of the graph [28].

The area under the ROC curve (AUC) is used as a single number which represents the ROC curve's shape. It is desirable that the AUC is more than half of the total area. It should therefore approach 1. The accuracy of classification as determined by the shape of the ROC curve is shown in Figure 2.3. This figure indicates that a *good* ROC curve lies in the top left area of the graph while a *worthless* curve is the same or below the positive linear function.

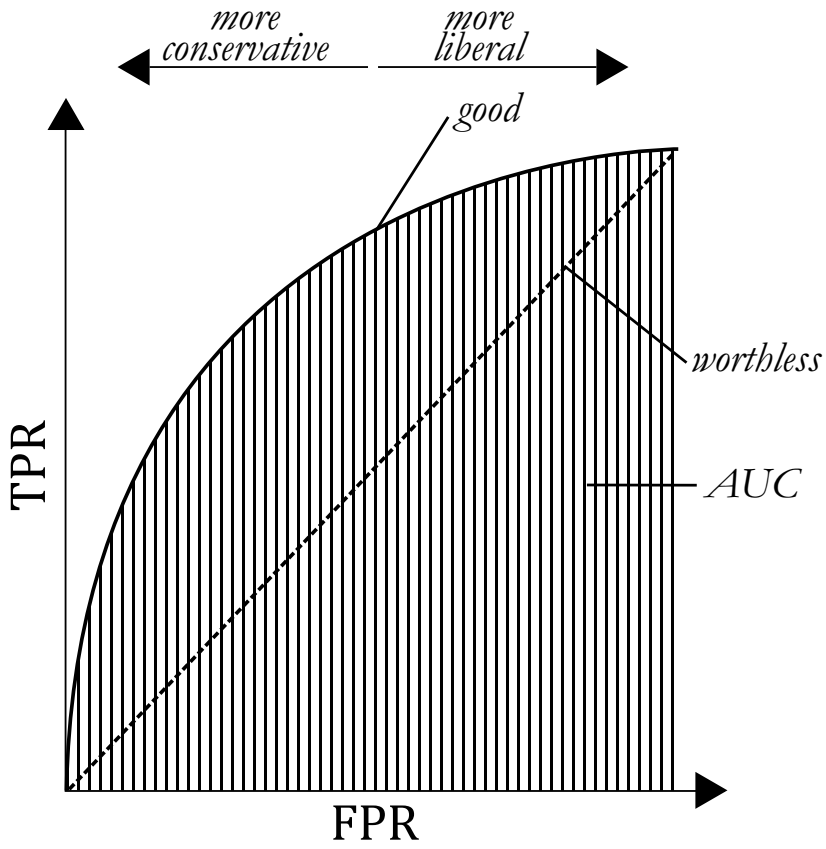


Figure 2.3: The ROC curve and resulting AUC.

Figure 2.3 also shows that the values of the TPR and the FPR can be either *conservative* or *liberal*. Conservative values indicate that the system is more likely to reject a legitimate user but less likely to accept an imposter. Conversely, liberal values indicate that the system will more easily let an imposter in but will have fewer false negatives on genuine users. Values in

the left-hand area of the ROC curve are conservative. They only classify a data instance as positive for very high confidence levels. Values in the right-hand area are liberal. They will easily classify data instances as positive. Liberal values are more desirable for usability, and conservative values are more desirable for high security applications.

The next section reviews previous research related to touch biometrics and analyses reported results using the measurements introduced in this section.

2.4 The Use of Biometrics in Smartphones

Biometrics show promise to alleviate the usability and security problems in authentication. Smartphones include an impressive array of sensors that could be used for biometrics, including accelerometers, video and audio recorders and global positioning system (GPS) tracking tools. Section 2.4.1 discusses the use of sensors other than the touchscreen for authentication on smartphones.

Section 2.4.2 reviews work done on hardening passwords using touch biometrics. Section 2.4.3 discusses the use of touch biometrics alone, as a continuous biometric.

The work summarised here was used to inform the decisions made regarding the development of the proposed touch biometric system, introduced in Chapter 3.

2.4.1 Sensor Based Biometrics on Smartphones

Many studies have successfully used biometric authentication on mobile phones using either additional sensors added to the device [29, 57] or using the built in hardware of the phone [2, 23, 24, 40, 46].

Derawi et al. [23] worked on recognising a user's gait style by using the lower quality accelerometers included in smartphones as opposed to dedicated accelerometers, video recordings or pressure plates that were used in the past. The authors concluded that the mobile phone accelerometers could achieve an EER of 0.2, which is 50% higher than the EER of systems using dedicated gait-recognising hardware. The authors concluded that the technology was not accurate enough to provide reliable authentication.

In a recent study Derawi and Bours [24] attempted smartphone accelerometer based gait recognition again. The system was improved by considering walking speed and different walking cycles and by using a variety of different classification methods. Their results were improved and a FAR of 0.014 and a FRR of 0.107 were reported. In this case, the threshold that gave

the smallest sum of FAR and FRR was used. Derawi and Bours concluded that these results were in an acceptable range of errors and that future work could focus on improving the accuracy by taking the activity the user was performing into account.

Ho et al. [40] reported an EER of 0 using a support vector machine (SVM) algorithm to recognise gait on a smartphone. The authors could only achieve these perfect accuracy rates when they had a ratio of more than 70% positive examples to learn from. Therefore, the amount of data collected from users is important to achieve high accuracy.

Agrawal [2] proposed a continuous biometric model based on facial recognition on an Android tablet. The face is captured and compared against a set of training images whenever certain applications are accessed. If the recognition fails and an imposter is detected, the tablet or the application is locked. The percentage correct was used as an accuracy measurement and was reported for arbitrarily chosen threshold values. It fell between 50% and 80% using two different facial recognition algorithms. Since the EER and AUC were not reported, the true accuracy of the system is not clear. The method of locking the tablet when an application is accessed is a good way to increase security without sacrificing the system's performance.

Some research has also focused on using network data for authentication or identification [46, 55]. Kuseler et al. [46] proposed using the data from a wireless internet network to verify the GPS location reported by a smartphone to harden authentication of online payment transactions. De Montjoye et al. [46] investigated how easily people could be identified using their network carrier data alone [55]. The research showed that four spatio-temporal points were enough to uniquely identify 95% of the individuals in their study.

The sensors on smartphones are informative enough to allow various methods of identifying users. This is promising since it indicates that a combination of different sensor-based authentication methods would be very accurate. This dissertation focuses on the use of the touchscreen for authentication. Future work could consider combining this with other sensor-based authentication methods as described above.

2.4.2 Touch Biometric Hardened Authentication

Many smartphones and other mobile devices, such as tablets, now use touch surfaces as a primary input method. Because of the popularity of this sensor, recent work has focused on using the touchscreen to harden traditional authentication mechanisms [8, 9, 22, 67, 68, 74, 90]. The touchscreen is used to augment the password, PIN or pattern authentication. Therefore, the user still needs to remember a password, PIN, pattern or gesture, but the

touchscreen is used to add an extra layer of authentication. Touch biometrics become an unobtrusive and silent part of a two-factor authentication system, improving security without reducing the usability. However, usability is not improved.

Sae-Bae et al. [67] developed a new approach to biometric authentication on tablets called "biometric-rich gestures". These are multi-finger gestures where the hand geometry and the dynamics between fingertips are used to authenticate users. This system achieved acceptable error rates at an EER of 0.1. The experiment in this dissertation also recorded multi-touch gestures, but did not require participants to enter a memorised gesture. Rather, the classification was done based only on the touch data, and not on verifying this data with an additional knowledge-based authentication mechanism.

Shahzad et al. [74] implemented a gesture-based authentication mechanism similar to that of Sae-Bae et al. [67], discussed above. This system mainly focused on the biometric features of the input, rather than the gesture itself. Shahzad et al. used Windows Phone 7, and focused on enrolling users and testing enrolled data using an application implemented on the phone. The system recorded seven gesture features: the velocity magnitude, device acceleration, stroke time, inter-stroke time, stroke displacement magnitude, stroke displacement direction and velocity direction. The experiment performed in this dissertation also included these features in the recorded touch data, except for the displacement features, as these are represented in terms of coordinates as discussed in Section 4.3. Shahzad et al.'s system achieved an EER of 0.05 using between 15 and 25 enrollment samples from the users. The experiment conducted in this dissertation recorded touch data from 30 participants.

De Luca et al. [22] used biometric information from the touchscreen to harden the pattern password mechanism in the Android OS using the dynamic time warping algorithm. The features analysed were the touch coordinates, pressure, size, time and speed. These features are also included in the experiment conducted in this dissertation as described in Section 4.3.

De Luca et al. conducted two studies – one in a controlled lab environment and another in the field over several days. The authors concluded that more data was required, including data on imposters who maliciously attempt to mimic touch behaviour. The analysis of the accuracy of the proposed touch biometric system in this study also investigates the effect of the amount of data in Section 6.4.1.

The research of De Luca et al. also showed that devices need to be kept consistent so as to eliminate differences in touch data introduced by using different hardware. For example, if a new phone is used with a different screen resolution, the coordinates of a touch event will differ from coordinates

that were recorded on another phone. The experiment performed in this dissertation was also conducted only on one phone, but the proposed touch biometric system could be applied on multiple devices and OSs.

In the study of De Luca et al. participants' behaviours changed over time and it was suggested that the reference set for a particular user should change over several successful authentication attempts to allow for changes in behavioural patterns. Finally, the authors concluded that a more intelligent algorithm is needed to process the data. The data is too complex and dynamic to be processed with simple algorithms. This dissertation uses two different classification algorithms to classify touch data and uses a sliding window to keep the training dataset up to date.

The method of hardening traditional authentication using touch biometrics was also investigated by Angulo [8]. Angulo's study achieved the best mean EER of 0.1 using a random forest classifier, which is a combination of different decision trees' votes on a decision. The experiment conducted in this dissertation investigated the accuracy of a single decision tree to classify touch data. The use of various and customised classification algorithms to improve performance could be investigated in future work.

Saevanee and Bhatarakosol [68] added finger pressure and other touch data to harden traditional keystroke dynamics features. The research showed that the pressure measurement plays an important role in the feature vector. The system achieved an EER of 0.01 using this method. The experiment conducted in this dissertation also recorded pressure as part of the touch features.

Zheng et al. [90] used tapping behaviour to harden the traditional PIN authentication on smartphones. The system reached an average EER of 0.036 using this method. The system collected data on the acceleration, pressure on the touchscreen, finger size and time of each key press and release. After collecting the data, outliers were removed based on exceptionally long pauses between keys. A long pause indicates that a user is still learning the PIN code and hesitated. The research showed that the acceleration of the finger when performing a gesture is proportional to the pressure applied to the touchscreen and that the tapping force on the screen is proportional to the finger size. The analysis of touch data given in this dissertation investigates the information gain of different touch features.

The system proposed by Zheng at al. used a one-class verification system. A threshold value was applied on a dissimilarity score to distinguish between the registered user and others. In one-class learning, the size of the touch area was the least informative – where it was used as the only measured feature, it yielded the highest EER. However, when all the features were combined and tested, the EER was between 0.0365 and 0.0734. The research showed that

mimic attacks did not reduce the dissimilarity score between the attacker and the genuine user enough, and that only the acceleration of a gesture was observably reproducible by the attackers. The authors attributed this robustness to the multiple and unrelated dimensions of the data, the resolution of the measurements (such as time which is measured in milliseconds) and physiological features that are difficult to mimic.

Zheng et al. also found that the position of the hand did not significantly affect the error rates. However, the participants' behaviour did change when they were tapping while walking. This is consistent to the work of Aviv et al. [9] who found that walking introduced noise into sensor data.

The experiment conducted in this dissertation recorded detailed touch data which provided a similar resolution and resulting robustness.

These works indicate that the field of touch biometrics is growing and that, as part of a two-factor system, touch biometrics achieve high accuracy rates. It is important to collect enough information on the enrolled users, as this has a high influence on the accuracy. The algorithm used to classify users has a high impact on accuracy. The research summarised here indicates that a high number of touch features do not need to be collected in order for the system to be accurate. Rather, it is important to focus on the most important features. Furthermore, it is difficult for an attacker to mimic the touch biometrics of an observed user, especially if multiple features are recorded and if those features have a high resolution.

The work discussed in this section does not improve the usability of traditional authentication mechanisms, since users still need to remember their gestures, PINs or passwords and touch biometrics are only used to harden the traditional mechanisms. This dissertation will focus on continuous touch biometrics, which use passwords only as a fall-back. Previous work in the field of continuous touch biometrics on smartphones is introduced in the next section.

2.4.3 Continuous Touch Biometrics

A number of studies have recently investigated continuous authentication using touch biometrics on smartphones [29, 32, 50].

One problem that arises in the investigation of touch biometrics is the manner of collecting such sensitive information from the operating system. Collecting touch data is blocked since it could lead to security problems. Previous work has found various workarounds to this problem. The study by Frank et al. [32] used a "mimic" interface to simulate the act of using the smartphone normally. Their system achieved low error rates. Two actions were investigated – swiping vertically and horizontally. Vertical swipes

were recorded by asking users to read one of three randomly selected documents. Horizontal swipes were recorded by asking users to compare two images and switch between images with a horizontal swipe action. Their system, therefore, does not represent the complete set of real-world continuous interactions.

Li et al. [50] used the device logs on the Android system for collection of touch biometrics. These low level logs output information directly from the touch device itself. This is an opportunity to collect touch information if the user has root privileges on the OS. They collected data from 75 participants on two different smartphones. The users were encouraged to use the phone in a normal way over several days. The experiment therefore simulated real-life touch interactions.

Feng et al. [29] augmented the smartphone's sensors with a digital sensor glove, recording hand movements in addition to touch data. Each participant performed three distinct gestures – swiping, pinching and spreading and dragging or drawing shapes. These gestures were kept separately in the database. The use of distinct gestures indicates that this system was tested on a limited range of touch data.

In contrast to the related studies described above where data was collected using mimic interfaces or external sensors, in this study, touch data was not collected using a mimic interface or an external sensor, but was rather collected using an application embedded into the operating system.

Frank et al. [32] collected a dataset of raw touch events, including the phone's ID, the user's ID, the document being read, time in milliseconds, touch action, phone orientation, x-coordinate, y-coordinate, pressure, size of the area covered and finger orientation. From this set of raw data, 30 features were extracted from each range of raw events into one gesture.

To simplify the problem down to a binary classification problem, the two classes that were created were the *user of interest* and *all other users*. They state that they "...then test the classifiers on data that has not been seen by the classifiers during training time". It is therefore inferred that verification was performed in this study. The study did not include multi-touch functionality.

In their study, Li et al. [50] divided their data into distinct training and testing sets, excluding training users from the testing sets. That is, this study tested verification over identification. The raw collected patterns were classified into different gesture types. Each gesture was assigned its own metrics. For example, a tap gesture has only three metrics – the average touch area, the duration of the gesture and the average pressure. Each type of gesture was then classified using its own classification module. The five proposed gestures were sliding right, left, down, and up and tapping, illustrated in

Figure 2.4.

Therefore, to classify all five proposed gestures the proposed system by Li et al. needed five classification modules. However, it was shown that the feature extraction process and the classification itself was not detrimental to the phone's performance. They suggested that touch data could be submitted to a central location using anonymous group messaging software. New patterns for training would only be downloaded when the classification algorithm needs to be retrained. Otherwise, authentication would be done using the existing model. A failover approach was also suggested, where the re-authentication module is supported with another established authentication mechanism such as a password.

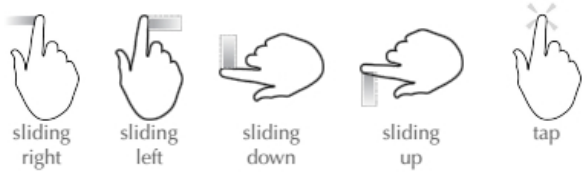


Figure 2.4: Common gestures identified in [50] (a reproduction of Figure 3)

Feng et al. [29] extracted gesture features from both the touchscreen and the sensor glove. Some participants used the sensor glove, while others did not. Collected touch gesture features included coordinates, direction of the motion, speed and distance between multi-touch points. Each gesture was divided into three parts, the beginning, the main motion and the end of the motion. To record data, Feng et al. used an application on Android that collected touch data. Participants were instructed to perform a set of touch tasks multiple times. It is therefore not a representation of real-world phone use. The system recorded several multi-touch gestures, including flick, pinch, spread, drag and rotate. These are illustrated in Figure 2.5. The system separated each kind of gesture into its own dataset. Their results were improved by using a sequence of gestures rather than one single gesture. This study did not focus on identifying which touch biometric features were most informative.

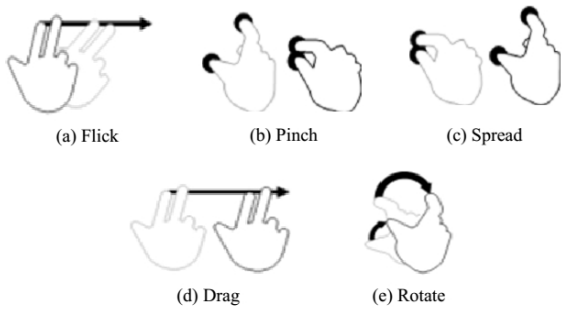


Figure 2.5: Multi-touch gestures identified in [29] (a reproduction of Figure 1).

To perform a complete investigation, the study presented in this dissertation tests authentication by both verification and identification. Furthermore, two types of datasets were created: the unprocessed raw data, and highly processed and feature extracted data.

Frank et al. [32] tested the data using the k-nearest neighbour (k-NN) and support vector machine (SVM) classification algorithms. The study showed EERs of 0 to 0.4, with the lowest error rates achieved using the SVM algorithm. They remarked that the error rates increased as the time between tests increased and concluded that user behaviour changes over time. They suggested that the accuracy could be improved by creating a multi-modal biometric system that takes into account, for example, location, accelerometer data and audio patterns.

In the study by Li et al. [50] the classification using the SVM algorithm also achieved relatively high accuracy rates for all classification modules. They concluded that finger pressure is not a distinguishing feature of touch interaction as it does not result in informative probabilities when classifying patterns. In other words, the finger pressure does not contain enough information to make a valuable contribution to classification. This is in contrast to the findings of Frank et al., given above. Furthermore, Li et al. suggest that tap gestures do not contain good metrics since these gestures occur too quickly to produce distinguishing features. However, the research mentions that the smartphones used in the experiment had low-end touchscreens. Therefore, some metrics may become more important as technology improves on touchscreens. Their system excluded unimportant metrics based on their analysis of the data, and would therefore need to be redesigned to be deployed on other phones. The amount of training data was also analysed, and the analyses concluded that a large training set improves performance up to a point, when overfitting occurs. Furthermore, the running time of the

feature extraction and classification also increases as the amount of training data increases. Therefore, training datasets should be limited in size to overcome processor limitations on smartphones.

Feng et al. [29] used a set of three classifiers – a decision tree, a random forest and a Bayes net classifier. Their system achieved a FAR of 0.047 and a FRR of 0.001. These rates are based on the assumption that if three out of seven gestures match the authorised user the user remains authenticated. The authors mention that their system "strives to achieve a low FRR".

Previous work focused on a variety of algorithms, and most included the SVM algorithm, a decision tree based algorithm or instance-based learning. In this study, the C4.5 decision tree algorithm and K* instance based classification algorithms were used to develop classifiers. The resulting accuracy rates are reported in detail and an analysis of EER, AUC and the corresponding ROC curves are discussed in Chapter 6.

2.4.4 Critical Analysis of Related Work

Related research on using behavioural and continuous biometrics for authentication on smartphones was restricted in the following ways:

Limited actions

The research investigated limited actions that users could perform, as is the case with a password hardening approach in [8, 9, 22, 67, 68, 74, 90]. Furthermore, these studies show that touch biometrics can achieve good accuracy, but does not attempt to investigate how such a system could be included in the OS. The proposed biometric system in Chapter 3 does not have these limitations – touch data during normal phone use is collected and the system is implemented and deployed as part of the OS.

Incomplete results

In some related work, the reported results are incomplete, such as unspecified threshold values for error rates [2] and limited error output values [50]. This dissertation gives a complete overview of error rates achieved and the threshold values used in Chapter 6.

Datasets not discussed

It is not clear whether the research was investigating verification or identification [29]. That is, it is unclear whether the system knows about all users, or whether some users are treated as unseen users. If the research does not test

the accuracy of the system against completely unseen data, the results may appear more accurate than the system would be in reality. This dissertation describes the datasets used clearly in Chapter 4.

Multi-touch not supported

Some research [32, 50] did not investigate multi-touch interaction. However, multi-touch is a mature technology that is associated with modern smartphones. Furthermore, if only one finger can be recorded at one time, this could leave a vulnerability in the system. An attacker could use multi-touch gestures to fool the system. Therefore, this dissertation investigates multi-touch interaction.

2.5 Conclusion

Section 2.1 discussed the problem of security vs. usability. In order for an authentication mechanism to be practically usable and beneficial in terms of protecting system assets, it needs to be both usable and secure. If an authentication mechanism is not usable, it will not be popular with users and they will find ways of circumventing the authentication mechanism [1, 25, 85]. If the authentication mechanism is not secure enough it will not present a sufficient barrier to attackers and will essentially be useless.

Section 2.2 introduced the different ways of authenticating users: by using something they have (tokens), something they know (passwords) or something they are (biometrics).

Tokens are very usable since the user simply needs to present the token for authentication. However, tokens are not very secure, since possession of the token allows access to the system.

Passwords require users to remember unique sequences of characters, thereby decreasing their usability. Users react by creating passwords that are not secure.

Biometrics are generally very usable because they rely on something that the user *is* and not something they can lose or forget. Biometrics are very secure as long as the implemented system has low error rates and any necessary extra features, such as liveness detection, are implemented. Continuous biometrics are more usable than traditional biometrics, as the user does not need to perform an action to authenticate and the system is constantly active. Continuous biometrics are more secure, since the system is constantly re-evaluating whether the user is genuine.

Section 2.3 described a collection of accuracy measurements that are applied to biometric systems. These measurements were chosen since they give a good overview of the system's accuracy as a whole as well as its applicability to different functions. In this dissertation, the proposed system is evaluated using the false accept rate (FAR), false reject rate (FRR), true positive rate (TPR), true negative rate (TNR) and equal error rate (EER), as well as the receiver operator characteristic (ROC) curve and the corresponding area under the ROC curve (AUC).

Section 2.4 summarised related work in using biometrics for secure and usable authentication on smartphones. Previous research showed that touch biometrics are accurate enough to be used as an authentication mechanism. However, the research still requires further investigations.

The following chapters introduce the novel contributions of this dissertation. The proposed solution is applied to form a model representation of this system to determine whether it is feasible. This model is described in the next chapter.

Chapter 3

Proposed Continuous Touch Biometric System

Smartphones need to be better protected by improving their authentication mechanisms. Currently, the protection of smartphones still relies on traditional authentication mechanisms such as passwords, PINs and patterns. These mechanisms place the burden of remembering on the users, who attempt to ease the difficulty of remembering their passwords by setting weak passwords or writing them down. Therefore, current authentication mechanisms have low usability and security. It is suggested that biometrics may improve authentication by removing the burden of remembering from the user. Furthermore, continuous authentication provides another level of authentication by creating a constant barrier to intruders. Continuous authentication keeps attackers from accessing the phone when it has already been unlocked or when the phone's user did not enable the password lock mechanism at all. That is, the period of vulnerability of the phone is reduced since authentication is done continuously, and not just when initially unlocking the phone.

To improve authentication on smartphones, this chapter proposes a model of authentication by augmenting a smartphone's operating system with a touch biometric system. The proposed solution focuses on using the Android OS, as this OS was used in the experiment. However, future work could implement this system on other smartphone OSs such as iOS or Windows Phone.

Section 3.1 discusses whether touch data could be used as a biometric characteristic. Section 3.2 discusses the requirements of a successful touch biometric system. Section 3.3 introduces the proposed design of a touch biometric system.

3.1 Touch Data as a Biometric Characteristic

As discussed in Section 2.2.3, a biometric characteristic should ideally be universal, permanent and collectable [42]. This section discusses whether touch biometrics are able to meet these goals.

Touch biometrics are *universal* on most modern smartphones. In the case where the smartphone does not have a touchscreen, a different biometric should be used such as keystroke dynamics. Similarly, touch biometrics would not work for a user who is not able to use a touchscreen. Therefore, touch biometrics are universal within the boundaries of any users who habitually use smartphones with touchscreens.

Touch biometrics are *distinctive*: The distinctiveness of touch biometrics have been shown by other authors as described in Section 2.4.2 and 2.4.3 [8, 9, 22, 29, 32, 50, 67, 68, 74, 90], and is also investigated in this dissertation. These studies show that touch biometrics are distinctive and can be used to identify different users.

Touch biometrics stay *permanent* for short periods, but may change over time. A touch biometric system should therefore be designed to cater for changing user behaviour by updating its training data and retraining its classification algorithms. A sliding window approach for the biometric feature database is proposed. This approach keeps adding new training data while discarding old training data from the database.

Touch biometrics should be *collectable*. Touch biometrics may be difficult to collect due to security restrictions on the smartphone's OS. For example, Android restricts the collection of touch data by unauthorised, non system-level applications. However it is possible to integrate the authentication system into the smartphone's OS. The authentication system will thereby be protected and will be difficult to be exploited by attackers.

3.2 Required Features

This section discusses features that are required for the proposed solution to be successful. These features are derived from the patterns that emerge from previous studies, as discussed in Section 2.4.

3.2.1 No Explicit Enrolment Phase

It is important to maintain usability of the system as a priority, to avoid users getting frustrated with the system. Therefore, users should not be required to explicitly enter a set of patterns or perform a set of tasks for

the system to be trained. A possible solution is to activate the enrolment phase in the background of the system, only alerting users when enrolment is complete. The point at which enrolment is complete could be defined as a static database size requirement, or it could be tested to reach a set of desired error rates.

3.2.2 Failover

A failover provides a backup for when the system has made an error and the genuine user has been locked out of their device. In the case where the matcher has detected several failures in a row, it should raise an alarm. This could revert the phone back to its traditional authentication mechanism to re-authenticate the user. This is to avoid a complete lock-out if the user has suddenly changed their behaviour. A change in behaviour could happen if the user is distracted, has switched hands, or is playing a new game. However, if the system reaches low error rates and is reliable in its accuracy, this failover could be removed, or the failover could be changed to a more usable authentication mechanism. For instance, a facial recognition system using the front facing camera on the phone could be used as a failover.

3.2.3 Customisability

Because there is a tradeoff between FAR and FRR, the threshold value at which users are locked out needs to be optimised to achieve a good balance between security and usability. A standard threshold (level of strictness) could be used as a default, and the level could be decreased or increased as the system becomes more accurate with more data.

However, users should be allowed to specify their preferred threshold value. Users who feel that they need added security could increase the threshold value, while users who prefer less lockouts could decrease the threshold value. The value could be bounded to keep users from setting the value too high or too low. Corporate security policies could, for instance, utilise this functionality if a smartphone contains confidential data. Conversely, private users could set the threshold value relatively low.

3.2.4 Embedded into the operating system

For the touch biometric system to work, it needs to be part of the OS that is installed on the phone. Consumer level applications do not have the required permissions to access touch data. Furthermore, it should not be possible for an imposter to uninstall the application. Users should, however, still be

allowed to disable the application if they feel that it constitutes an invasion of privacy. To allow this, the failover authentication mechanism could be used to verify that the legitimate user is disabling the application. Ideally, it should be impossible for an attacker to reach this point, since the system should lock once the attacker attempts to use the phone.

3.2.5 Support for Multi-Touch Gestures

Modern touchscreens usually support multi-touch gestures and any touch biometric system should also consider multi-touch to achieve accurate results. If multi-touch gestures are not recorded, some users will have lower accuracy if they use a lot of multi-touch gestures. Furthermore, if only one finger can be recorded at a time it could leave a vulnerability in the system. Imposters could use multi-touch gestures to circumvent the system.

3.2.6 Continuous Learning

It should be possible to add new training data to ensure that the system is up to date with its owner's touch behaviour. It should be relatively easy to add new training data, and it should not require a resource-intensive process to occur on the device whenever retraining is needed. There should also be no need to first manually pick out important features. The matcher module should be capable of distinguishing important features from others. This is discussed in Chapter 5.

3.2.7 Adaptability and Device Independence

Each device that runs Android needs to implement unique drivers to support its specific hardware such as its sensors and input devices. Therefore, each smartphone's version of Android is slightly different, since each version of Android needs to include drivers specific to the hardware on that smartphone. Furthermore, new versions of Android are released periodically. Therefore, each device could have a newer or an older version of Android.

The implementation of a touch biometrics system could depend on whatever measurements are made available by a particular device's touchscreen. For instance, a more advanced smartphone would have more accurate values for pressure readings than a mid-range phone.

It is therefore important to note that different devices and versions of Android have different capabilities and that a proposed biometric system should be adaptable to any device. This is also true for different versions of the same OS. The core system could be kept constant for most of these situations, but

the format of the collected data would definitely change. Therefore, the training data for each device and OS combination should be kept in a separate database. This would ensure that the data is not skewed by a slight difference in hardware or software.

Adding new sensors to the proposed system should also be relatively easy. As smartphones are improving, new sensors are being added to them. Recently, new devices have been launched, which include temperature and humidity sensors [26]. If a new device implements new touch interfaces, the system should be able to add any other sensor data to the feature vector.

3.2.8 Independent Operation

The proposed system should not rely on a constant internet connection to be able to function. It should also not depend on any extra hardware, except the embedded sensors on the smartphone.

3.3 System Design

The proposed system consists of the biometric system modules as described in Section 2.2.3, namely sensor, feature extractor, matcher and database [42]. This basic framework is extended to include a failover component and a re-training module. The database is extended to include both local and remote components. The proposed system is intended to be very adaptable. The framework would allow the addition of any sensor and any algorithm in the matcher module. The design of the biometric system is illustrated in Figure 3.1.

3.3.1 System Modules

The *sensor* records any touch data that is available to be recorded by the device and its operating system. The recorded data will differ between different versions of Android and different devices. However, it will be consistent within these boundaries. The Android API exposes a set of functions that record touch information. For devices that do not provide these details, the information will often remain static or be inferred from other features. The same system can therefore be used on many different versions of Android. Therefore, if a user has the same smartphone and Android version as another user, their data will be represented similarly. The sensor sends the recorded data to the feature extractor module.

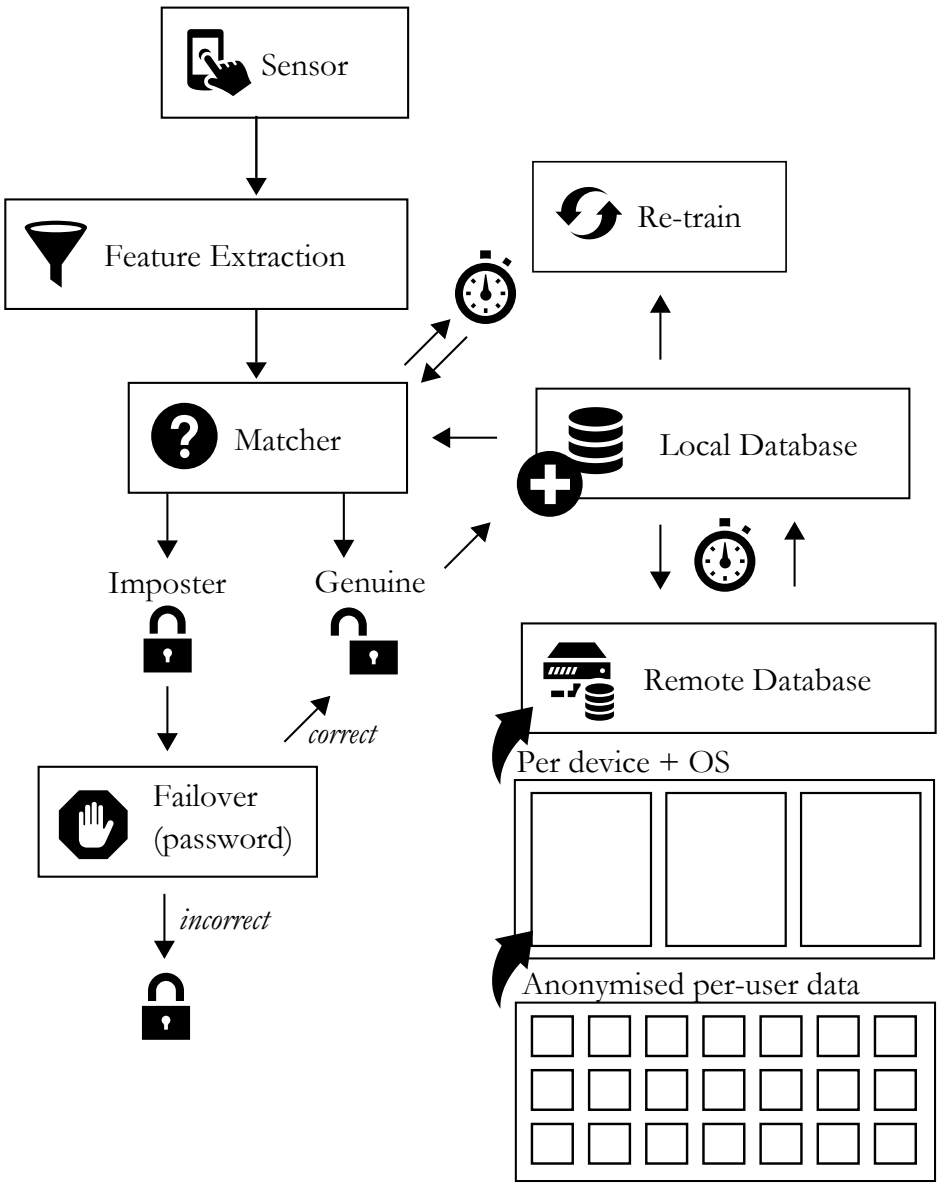


Figure 3.1: An overview of the proposed system.

The *feature extractor* translates the touch data collected from the sensor into a known format that can be understood by the system. The module filters and transforms data to ensure the required features are present to be sent to the matcher module.

The collection of the touch data from the sensor and the extraction of touch data features on the proposed touch biometric system are discussed in Chapter 4.

The *matcher* is a machine learning module that makes decisions based on data it has been trained on and the data being presented to it currently from the feature extractor. Any classification algorithm could be used here. The matcher module makes the decision whether each new data instance from the feature extractor represents a genuine user or an imposter. The decision is a binary value that is sent along with the data from the feature extractor to the database. Therefore, the threshold value for setting the level of strictness of the matching would be configured here.

The *database* stores the touch data of the current user locally. In a remote database data is collected from other users using other similar devices that run the same version of Android. The similar device and version of Android ensures that the remote and local databases will be compatible in terms of the kind of touch biometric features that are collected. The database will contain data from the phone's user and will collect a set of data that represents "not the user" from the remote database. This data will be used to train the matcher module for classification. The details around this "imposter" dataset are discussed in Section 4.4.2.

The accuracy in matching and the storage of touch data for the proposed system is analysed in Chapter 6.

The remote database should ensure that the data is de-identified by assigning unique user IDs to patterns that cannot be tracked back to a specific user by an attacker. Only the local data needs to be associated with a specific user. The remote touch database is simply a collection of touch data that is used to represent "not the user" within the authentication system. Therefore, the data can be de-identified to keep attackers from being able to use data from the remote database to gain access to a specific user's phone.

3.3.2 The Continuous Authentication Process

To start using the biometric system, the system initiates a training phase. The training phase should be as unobtrusive and continuous as the rest of the authentication process. If the phone has been reset (wiped) or if it is the first time a user has accessed the phone, the initial training phase begins, if the user agrees. This phase assumes the user does not give the phone to anyone else for a set period of time. The user is notified that training has started. However, if someone else uses the phone during the initial training phase, the user could restart the training phase using the system's settings. During this phase, the user just uses the phone normally, while the system starts to collect touch data. Once an acceptable level of error is achieved or if a pre-determined time has elapsed, training stops and the continuous authentication system activates. The acceptable level of error could be set

to a static value at this point. For example, if the EER reaches 0.05, the continuous authentication system will be activated. If the EER stays high even after the training time has elapsed, users could be prompted whether they want to activate the system anyway. Once again, the user is notified that training has stopped and that the continuous authentication is now active.

The local database contacts the remote database and receives a selection of random touch data from other users of the same device and OS. This collection serves as the "imposter" dataset. The local dataset sends the collected data from the genuine user to the remote server, to be used as part of the imposter dataset for other users. This process of communication between the two servers occurs intermittently while the system is in operation. As it is not necessary to get new imposter information frequently, this exchange could be done infrequently. For example, once a day the new touch data is uploaded and the imposter dataset may be updated.

When a user touches the screen and the sensor module collects new data, each data instance is sent to the feature extraction module. The feature extraction module takes the raw touch data and places it in a format that is required by the matcher module. During the matching phase, the module compares new touch data to the stored data from the genuine user and other users inside the local database module. If the data instance matches the profile of the genuine user, the new data instance is also added to the local database. An older pattern is removed for each new pattern to save storage space and keep the matcher up to date. If an imposter is detected, the systems goes into failover mode and a password or other authentication mechanism is requested. If the password is correct, the user is assumed to be genuine and the data instance that caused the failure is also added to the local database. If the password is incorrect, the user will not be able to access the system and the data instance that caused the failure is not added to the local database.

The matcher module periodically undergoes a re-training phase to incorporate new data instances into the matcher's model. The point at which the matcher module re-trains could be done after a set amount of time has passed, or a set amount of data has been added. While re-training is occurring, the matcher module should not become non-operational. The existing classification model could be used while the new model is being trained. This re-training is only necessary if the classification algorithm used in the matcher module relies on a pre-built model for classification. For example, a neural network will need to incorporate new training patterns since there is a distinct training phase when data patterns are used to build the network. For a time, the neural network operates without updating the network, and simply makes decisions based on the pre-built network. Some classifiers do

not require this extra step. For example, lazy classifiers will just compare each new pattern to the set of training patterns each time a new pattern is presented to the classifier. Therefore, these classifiers do not need to update their model. This dissertation tests the system using both types of classification algorithm (pre-built model based classification and lazy classification). Classification algorithms are described in more detail in Chapter 5.

3.4 Conclusion

This chapter discussed the requirements of a continuous touch biometric system and introduced the proposed model for such a system.

The next chapter starts the discussion on the setup and design of the experiment by giving an overview of the data collection phase. The experiment is a controlled representation of the system and investigates whether the proposed system could be implemented successfully.

Chapter 4

Touch Data Collection and Feature Extraction

The touch biometric system as proposed in Chapter 3 is a theoretical framework describing the features of the system. To investigate whether this framework could be implemented and how accurate the system could be, a small scale model of the larger system was implemented, including the central functions of data collection, feature extraction and matching.

This chapter discusses the specifics of collecting the data and extracting features from it (the first two modules). Specifically, it investigates how a touch biometrics system could be deployed onto an Android phone and which touch features could be collected.

To collect the data from the smartphone, a tool was needed that could record touch events. The implementation of this tool is discussed in Section 4.1. The tool was used to collect the touch behaviour of 30 participants while performing six common tasks on their smartphones. Section 4.2 describes the instructions given to participants during the experiment. The data from the participants was filtered and aggregated to form the datasets.

To investigate which features are important for touch interaction, two types of datasets were created. The first contains the raw data from the Android *motion events*. The second contains features extracted from the raw data to form gesture data. The features contained in the raw and gesture datasets are discussed in Section 4.3.

To investigate both the identification and verification of users, two additional datasets were created for each task. The identification dataset contains all the touch data that was collected from all the participants. The verification dataset contains touch data from the genuine user and a set of randomly chosen anonymised data from all the other users. Identification and verification datasets were created for *both* the raw and the gesture data. The creation

of the identification and verification datasets is discussed in Section 4.4.

4.1 Tool Implementation

The smartphone used in the experiment was an HTC Desire. This phone was chosen to represent an average smartphone and was not the newest on the market. The Android OS was chosen to perform this study as it is open source and easy to modify. The version of Android running on the phone (Android 2.2) was replaced with Cyanogenmod 7 – a custom derivative of the Android OS developed by a community of independent developers. It enables users to replace the stock Android OS running on the phone. Cyanogenmod 7 is based on Android 2.3, a newer version of Android that is not available officially on the HTC Desire. At the time of the experiment this was the newest version of Cyanogenmod that was available for the HTC Desire. Without using Cyanogenmod, the phone's version of Android would have been too old to perform the experiment.

The Android SDK provides a set of motion event features that can be collected to form datasets for touch biometrics. A motion event is triggered whenever a user touches the touchscreen. It is therefore a good way to collect touch biometric data. The motion event contains information such as the pressure applied to the screen or the coordinates of the touch. The details of the collected event features are described in Section 4.3.

The approach used to collect touch event data was to create an application that overlays a transparent screen (or *view* in Android) over all other applications and eavesdrops on their touch events. However, Android does not allow applications to do this, since it could be exploited to write keylogger applications. This restriction in gathering touch data from the system has also been shown in other research by Frank et al. [32], Li et al. [50] and Feng et al. [29], where either the system logs were snooped to collect data, or a mimic interface was created to collect data as is discussed in Section 2.4.1.

Only the foreground application that is currently being used is allowed to record detailed touch information. Any application that is overlaid on top of the foreground application is blocked from recording this information. Views with their layouts specified as type *system overlay* may only receive limited touch data [7], but views with their layouts specified as *secure system overlay* are allowed to collect any touch information. However, the type *secure system overlay* can only be assigned to system applications and needs special permissions.

Android needs to grant permission to the application to use the *secure system overlay* type of layout. Therefore, the OS needed to be altered to

allow this application to collect information about touch events.

Android installations usually contain a set of developer tools. Among these tools is an application called "Pointer Location". When activated, this application displays an overlay view on top of other applications and displays the touch data features that it can measure. Pointer Location's window layout is of type *secure system overlay*. Pointer Location outputs data to the system logs, but its output is hidden and cannot be read directly from the logs.

To collect touch data, the stock Pointer Location application was replaced with an altered version for the experiment. The code to record touch data was therefore hidden within this application. This altered version recorded more features and revealed its output to the system log. In a real-world implementation, the data would rather be stored to the phone's internal database. The application was also made completely invisible to the users, so that it did not display any indication that it was running. The altered OS was compiled and deployed to the phone. Once the Pointer Location application was enabled on the new OS, it could be used to collect any required touch data.

The tool collected every motion event that was triggered by the operating system. Because it had the required permissions, it was capable of recording any touch data. The raw data recorded by the tool was filtered into different datasets, described in Section 4.3. The next section briefly describes how touch data was collected from the experiment participants.

4.2 Experiment Protocol

To collect the touch event data from the participants, the following steps were followed. At the start of each session the nature of the experiment was explained and each participant was asked to read a disclaimer, sign an informed consent form and to fill in a brief questionnaire. The questionnaire contained questions about the participants' experience with smartphones and their field of expertise. Most of the participants were from the IT field and had average to high experience in using smartphones and other devices such as tablets. Finally, they were directed to perform six tasks on a smartphone that represented real world interaction with a phone while their touch data was being recorded. These six tasks were:

1. Enter a pattern to unlock the screen. This pattern was kept constant for all participants.

2. Type a message and send it. The message varied between participants. The text was taken from poems by Dr Seuss.

3. Browse to Wikipedia and read the featured article. Participants were instructed to skim through the text as if they were looking for specific information.

4. Open a news app and read a story.

5. Play a game of Word Search and find five words.

6. Scroll through all the images in the image gallery and zoom in and out. There were 24 images in the gallery.

Once these tasks were completed, participants were shown a sample of their data if requested.

The touch event data of 30 participants was collected. The next section discusses how the touch event data from these participants was filtered into different datasets.

4.3 Raw and Gesture Datasets

Two datasets for two usage scenarios were created. The first dataset contained the raw data from the Android *motion events*. The second dataset contained features extracted from the raw data to form gesture data. A gesture is made up of several raw motion events occurring between a down and and up *action*. This is shown in Figure 4.1.

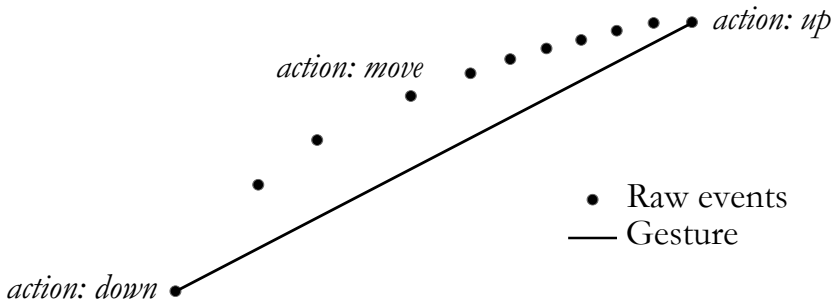


Figure 4.1: Raw events between a down and an up action are combined to form a gesture.

There is a special case for a multi-touch gesture. A multi-touch gesture is recorded as multiple gestures with a pointer ID assigned to each finger

that was touching the surface. The first finger that touched the surface has a pointer ID of 0, and each subsequent finger increments the pointer ID. For example, a "pinch"" gesture as shown in Figure 2.5 is recorded in the datasets as two scroll gestures with pointer ID's 0 and 1. This is shown in Figure 4.2.

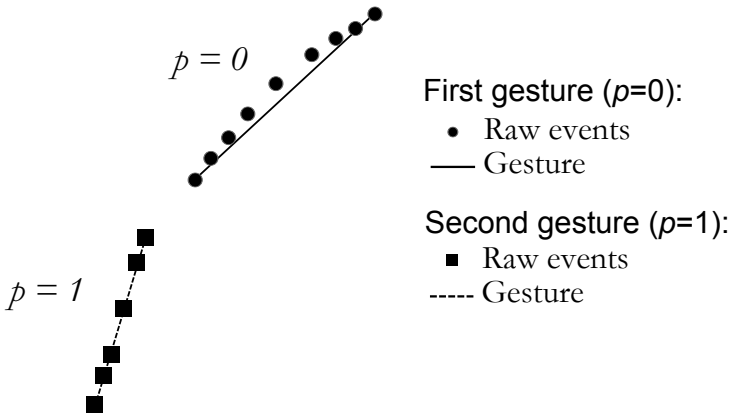


Figure 4.2: Multi-touch gestures are represented as two distinct gestures identified with a pointer ID.

The features contained in each dataset are listed in Tables 4.1 and 4.2.

Table 4.1: Raw features

Symbol	Description	Measurement
g	*Type of gesture*: The incomplete gesture that the user was busy executing at the point that this *motion event* was triggered.	Tap, double tap, scroll, fling or unknown
t	*Total time*: The time since the initial down action until this motion event was triggered. Resets after each up or cancel action.	Milliseconds
a	*Action type*: The action that was being performed on the screen.	Move, up, down or cancel
i	*Pointer ID*: The pointer ID that is performing this raw event.	0 or more

Continued on next page

Table 4.1 continued

Symbol	Description	Measurement
x, y	*Coordinates*: Most recent x and y coordinates of event.	Integer: > 0 and $<$ screen dimensions
p	*Pressure*: Finger pressure	Continuous: $0 - 1$
s	*Size*: Approximate touch area	Continuous: $0 - 1$
f_a, f_b	*Touch major and minor*: The length of the major and minor axes of the ellipse that describes the touch area	Continuous: Device dependent
f_c, f_d	*Tool major and minor*: The length of the major and minor axes of an ellipse that describes an estimation of the actual size of the finger or pen.	Continuous: Device dependent
f_θ	*Tool orientation*: The orientation of an ellipse that approximates the actual finger or pen.	Radians clockwise from vertical.
s_θ	*Screen orientation*: The orientation of the screen.	Landscape or Portrait

Table 4.1: Raw features

Gesture features listed in Table 4.2 are calculated using the vector of the movement from the start coordinates to the end coordinates. Other gesture features are means and standard deviations of raw features. Where the measurement of the feature is a nominal value, the value at the end of the gesture is used.

Some features were derived from the raw data for the gesture data. For instance, the inter-stroke time was derived from the times recorded between the last gesture's up action and the current gesture's down action. The vector features refer to the vector between start and end coordinates of the gesture.

Table 4.2: Gesture features

Symbol	Description	Measurement
g_{end}	*Type of gesture*: The complete gesture that the user executed when the gesture is ended. This could still be unknown if the system could not determine the type of gesture from the available gesture data.	Tap, double tap, scroll, fling or unknown
$i_{gesture}$	*Pointer ID*: The pointer ID that performed this entire gesture.	0 or more
t_t	*Total time*: Time from the first down action to the last up action.	Milliseconds
t_i	*Inter-stroke time*: Time between the beginning of this gesture and the end of the previous gesture	Milliseconds
x_s, y_s	*Start coordinates*: x and y coordinates of the start of the gesture.	Integer: > 0 and $<$ screen dimensions
x_e, y_e	*End coordinates*: x and y coordinates of the end of the gesture.	Integer: > 0 and $<$ screen dimensions
$\overline{x}, x_\sigma, \overline{y}, y_\sigma$	*Overall coordinates*: Mean and standard deviation of x and y coordinates over entire gesture.	Continuous: > 0 and $<$ screen dimensions
$\overline{p}, p_\sigma$	*Pressure*: Mean and standard deviation of finger pressure over entire gesture.	Continuous: $0 - 1$
$\overline{s}, s_\sigma$	*Size*: Mean and standard deviation of approximate touch area over entire gesture.	Continuous: $0 - 1$

Continued on next page

Table 4.2 continued

Symbol	Description	Measurement
$\overline{f_a}, f_{a_\sigma}, \overline{f_b}, f_{b_\sigma}$	*Touch major and minor*: Mean and standard deviation of the length of the major and minor axis of an ellipse that approximates the touch area.	Continuous: Device dependent
$\overline{f_c}, f_{c_\sigma}, \overline{f_d}, f_{d_\sigma}$	*Tool major and minor*: Mean and standard deviation of the length of the major and minor axis of an ellipse that approximates the actual finger or pen.	Continuous: Device dependent
$\overline{f_\theta}, f_{\theta_\sigma}$	*Tool orientation*: Mean and standard deviation of the orientation of an ellipse that approximates the actual finger or pen.	Radians clockwise from vertical.
$s_{\theta_{end}}$	*Screen orientation*: The orientation of the screen at the end of the gesture	Landscape or portrait
The following features are calculated using the vector of the movement from the start coordinates to the end coordinates.		
m_θ	*Vector angle*: The angle of the deviation from the closest *direction* (up, down, left or right).	$0° - 45°$
m_d	*Vector direction*: The general direction of the motion vector.	Up, down, left or right
$\|m\|$	*Vector length*: The length of the motion vector.	Integer: > 0

Continued on next page

Table 4.2 continued

Symbol	Description	Measurement
$\overline{m_v}$	*Vector speed*: The average speed of the motion vector.	Continuous: > 0
$\overline{m_a}$	*Vector acceleration*: The average acceleration of the motion vector.	Continuous: > 0

Table 4.2: Gesture features

The raw feature vector is defined in Equation 4.1.

$$(g, t, a, i, x, y, p, s, f_a, f_b, f_c, f_d, f_\theta, s_\theta) \tag{4.1}$$

The gesture feature vector is defined in Equation 4.2.

$$(g_{end}, i_{gesture}, t_t, t_i, x_s, y_s, x_e, y_e, \overline{x}, x_\sigma, \overline{y}, y_\sigma, \overline{p}, p_\sigma, \overline{s}, s_\sigma, \overline{f_a}, f_{a_\sigma}, \overline{f_b}, f_{b_\sigma}, \overline{f_c}, f_{c_\sigma}, \overline{f_d}, f_{d_\sigma}, \overline{f_\theta}, f_{\theta_\sigma}, s_{\theta_{end}}, m_\theta, m_d, |m|, \overline{m_v}, \overline{m_a}) \tag{4.2}$$

The raw dataset contained 14 unprocessed features, while the gesture dataset contained 32 features, excluding the user ID. The raw dataset was larger than the gesture dataset by a factor of seven. Therefore, on average, one gesture is made up of around seven raw events. The raw dataset is more computationally expensive to process than the gesture dataset, since there are more data patterns per gesture.

4.4 Identification and Verification Datasets

The raw and gesture datasets were used in two different ways to construct datasets for the identification and verification classification tasks. The first dataset contained the data of all experiment participants, and was used to test the identification accuracy of the biometric system. The second was a generated set of files per user, used to test the verification accuracy of the biometric system.

4.4.1 Identification Dataset

The identification dataset was a composite of all the users' data in one file. This dataset was created to test the identification capabilities of the bio-

metric system. It simulated a situation where all users are present in the database module of the system as described in Section 2.2.3. Therefore, the classification for matching was between 30 distinct classes.

In the experiment, this dataset was tested using 10-fold cross validation. Cross validation is a process where the dataset is divided into 10 equally sized parts. Nine of the parts are used to train the classifier, and the last part is used for testing. This process was completed 10 times using different parts for training and testing, and the average accuracy rates over the 10 runs was reported.

Cross validation was used here because there was no explicit test set. Ten folds were chosen as it is considered to be a good balance between the variance in measurements and bias introduced by smaller folds [45].

4.4.2 Verification Dataset

The second dataset was a collection of 30 training and testing sets for each user, created to test the verification capabilities of the biometric system. This task represents the case where the user's ID is known (only one user is registered on the phone) but needs to be verified through the use of biometrics.

Classification algorithms require a set of data on which to build the classifier (the training set), and a set of data used to test its generalisation abilities (the testing set). In this case, the difficulty lies in generating training data that does not give the classifier clues that it would not have in the real world. The classifier needs to learn a pattern for both the authorised user and the unauthorised users (everyone else). It is very unlikely that an unauthorised person's touch profile would have been recorded previously to train the classifier. Therefore, a training set should represent both the known user and a set of unknown users, but the testing set should not contain the same unknown users as the training set.

To create a verification dataset for all users, each user was in turn treated as the known user, and a set of other users as the unknown users. Therefore, in the dataset, the known user's touch data was labelled as their assigned user ID, but the other users' touch data was labelled as unknown. This set represented the scope of any other never before seen users. To test whether the system could detect an unknown user without having seen their data before, the set of unknown users used for training and testing was distinct. Therefore, if the classifier was trained on user 1, 3 and 5 as the unknown users, it would be tested on user 2 and 4 only. This rules out the possibility that the classifier had learnt the patterns for 1, 3 and 5 specifically rather than learning a generalised rule for unknown users.

For every user, a collection of 30 training and testing sets was created

from both the raw and the gesture data. These 30 files represented various combinations of unknown users used for training and then another set of users used for testing. For instance, if the first training file contained user 1, 3 and 5 as unknown training users, and 2 and 4 as unknown testing users, the next training file will contain 1, 4 and 5 as unknown training users and 2 and 3 as unknown testing users. All users were represented as part of either the testing set or the training set.

Each training set contained 70% of the genuine user's data and was tested using the other 30%. This ratio usually results in high accuracy when testing, but still avoids overfitting [27].

Formally, each training set included set A and set K_{train} and the corresponding testing set included set B and set K_{test} where:

K is the set of known user's data.
K_{train} is a randomly selected subset of 70% of the known user's data.
$K_{test} = K - K_{train}$ is 30% of the known user's data.
N is the set of all users' data excluding the known user's data.
A is a random subset of N, the same size as K_{train}.
B is a random subset of N, the same size as K_{test}, such that A intersection B is the empty set.

4.5 Conclusion

The processes described in this chapter were followed to collect and create datasets for identification and verification using both raw data and derived gesture data.

To reject imposters or authenticate genuine users, biometric systems require a matcher module, as described in Section 2.2.3. The matcher module is often implemented as a classification algorithm. Therefore, before the data can be analysed for authentication it needs to be processed using classification algorithms. Two classification algorithms were chosen to perform the analysis. These are discussed in the next chapter.

Chapter 5

Classification of Biometric Features

Classification algorithms are used to make decisions inside the matcher module of the touch biometric system. Classification algorithms are a form of supervised machine learning – they learn by building up knowledge based on labelled, known data. Specifically, the algorithms learn to recognise known patterns during *training*, using a set of data that is already labelled (the *training set*). Then the accuracy of the algorithms is tested using data that has not been seen by the algorithm (the *testing set*). The training set is usually larger than the testing set to give the classifier enough information to be able to effectively predict the class of data instances in the testing set.

The two algorithms used are the C4.5 decision tree algorithm [64], and the K* algorithm [17]. Both of these algorithms are provided as part of the University of Waikato's Weka data mining software [38]. These were chosen as they are relatively simple and fast, but performed reasonably well in initial tests. Furthermore, they represent both *model-based* (eager) learning and *instance-based* (lazy) learning. Eager classifiers build up a model from known data, and then simply test new instances against the model. Lazy classifiers store training data and when a new pattern is presented, they test the pattern against all the stored data instances using a set of learning rules. That is, eager classifiers take a long time to build a model, but classify quickly, whereas lazy classifiers do not build a model, but take longer to classify patterns. Eager classification is useful where speed is needed during classification, while lazy classification allows easy addition of new training patterns to the classifier.

The C4.5 algorithm is discussed in Section 5.1 and the K* algorithm is discussed in Section 5.2. Section 5.3 reviews the parameters and threshold values used in the experiment. Section 5.4 defines the acceptability of error

rates for the experiment.

5.1 C4.5

A decision tree algorithm constructs a data structure where the leaf nodes represent the decisions and the parent nodes represent different data features. Decision trees are eager classifiers. To classify a new instance using the modelled tree, the tree is traversed from the root node downward. At each node, the tree branches are followed according to the value of the data feature.

The C4.5 classification algorithm, defined by Quinlan [64] is based on the ID3 algorithm, defined by the same author [66]. The ID3 algorithm uses *information gain* to create the nodes. Information gain is calculated using entropy, the measure of the uncertainty in the dataset [75]. The calculation of entropy is [75]:

$$H(S) = -\sum_{x \in X} p(x) \log_2 p(x) \tag{5.1}$$

Where:
$H(S)$ is the entropy of S,
S is the set of all instances in the dataset,
X is the set of all classes in S and
$p(x)$ is the proportion of the number of elements from S in class x.

Information gain measures how much entropy was lost after the dataset was split on an attribute [18]. In other words, if the attribute is used to split the dataset into groups, do these groups have less entropy after the split? Does the split reduce the chaos of the sets and are they more uniform within their new groups? Information gain is therefore specific to each attribute in the dataset. Information gain is calculated in Equation 5.2:

$$IG(A) = H(S) - \sum_{t \in T} p(t) H(t) \tag{5.2}$$

Where:
$IG(A)$ is the information gain of attribute A,
$H(S)$ is the entropy of the set S,
T is the set of subsets created by splitting the set S on the attribute A,
$p(t)$ is the proportion of data instances from S in t, and
$H(t)$ is the entropy of subset t.

ID3 starts with all training dataset instances in the root of the tree. Then it calculates the information gain of each attribute that has not yet been used in the tree. The attribute with the most information gain is used to create the next split in the tree. The algorithm is recursive, stopping when all the data instances are of the same class, when all the data features have been used, or when some specified minimum number of instances is reached.

C4.5 offers several improvements over the ID3 algorithm. It can handle continuous attributes as well as discrete attributes and is capable of dealing with missing attribute values [65]. Furthermore, the information gain ratio is used instead of information gain. The information gain ratio is the ratio between information gain and the attribute's *intrinsic value*. This biases the tree against considering attributes that are "too distinct", thereby reducing the chance of overfitting on the training data. The final tree can also be improved by pruning the tree to remove branches that do not add to the tree's accuracy [64].

The J48 implementation in the WEKA toolkit [31] of the C4.5 algorithm requires two user-specified variables – the confidence factor c and the minimum number of instances per leaf m. The confidence factor c is used for pruning the tree. Smaller values for c cause the tree to be more pruned. The minimum number of instances per leaf m indicates how many objects should be present in a leaf to prevent the algorithm from splitting small datasets further. Smaller values of c and larger values of m therefore result in smaller trees.

Figure 5.1 shows the decision tree generated from a simplified version of the gesture touch dataset. The data is reduced to only consider mean pressure and coordinates, and is used to verify one user. That is, the classes are either "me" or "not me", or legitimate user and imposter, respectively. Each node splits the tree into two child nodes. The first split occurs on the attribute that has the most information gain. In this tree, pressure had the most information gain and therefore becomes the first split in the tree. This process continues recursively for each subtree – the next split in the left subtree is the x-coordinate.

Using the full dataset, the tree structures generated from the touch data are much more complex. In essence, the tree represents a series of decisions based on the value of each individual attribute. For example in Figure 5.1, if the pressure is higher than 4.3 and the x-coordinate is greater than 110.96, the tree predicts that the pattern belongs to the legitimate user. Decision trees are a simple method to classify data patterns, but shows good accuracy rates in the experiments. The results are discussed in the next chapter.

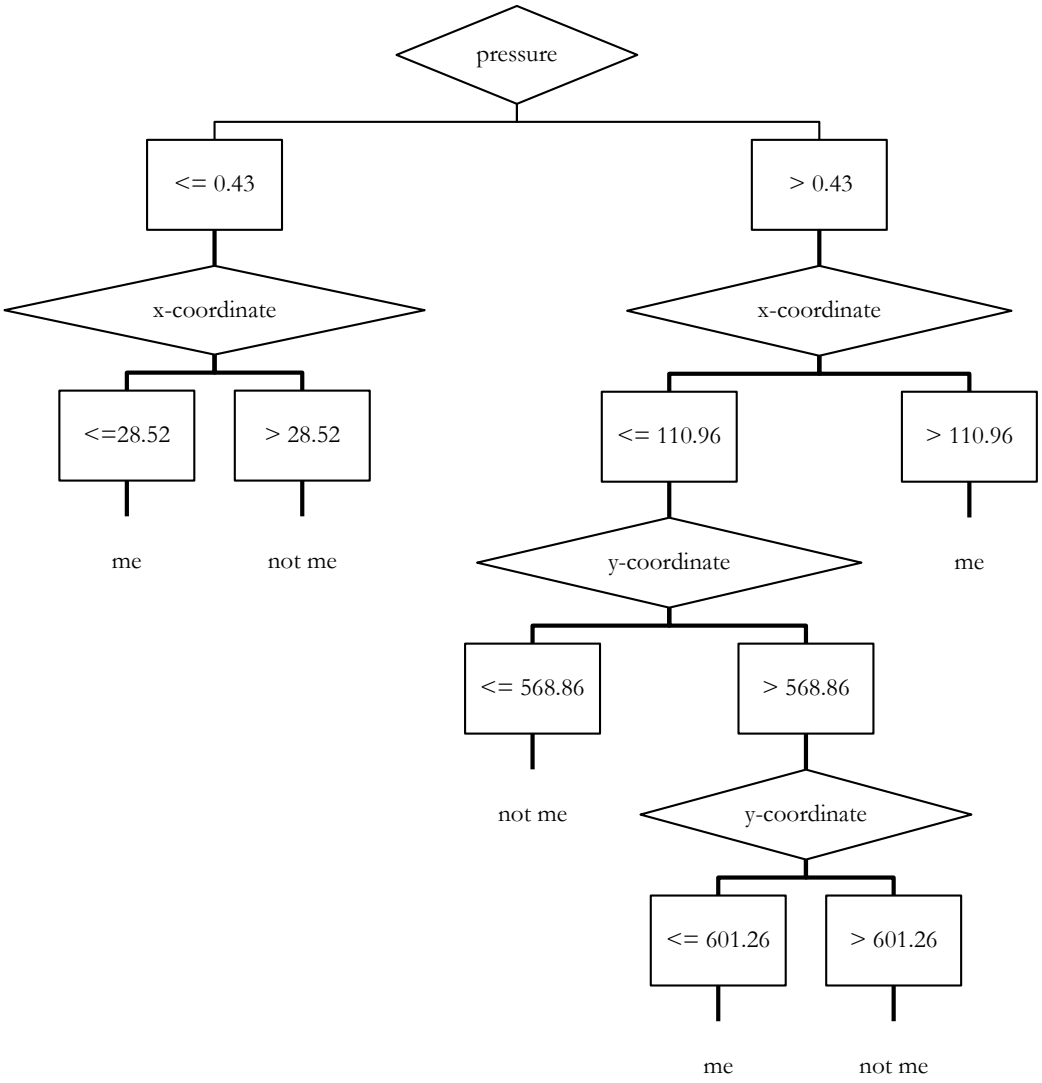


Figure 5.1: C4.5 tree example using a simplified version of the gesture touch data.

5.2 K*

The K* algorithm is a lazy classification algorithm, defined by Cleary and Trigg [17]. It has a set of training instances, but does not build a model before classification. As each new instance is presented to the algorithm it is classified by comparing it to all the training instances. Once it is classified it is added to the set of training instances.

The K* algorithm is therefore well suited to classifying continuous biometric data, as the training set can constantly be updated with a sliding window of new information. This sliding window can have a fixed size, or the algorithm could set some maximum size at which point the sliding window becomes active. However, it is slower during classification, since it needs to perform the necessary calculations in real time and cannot use a pre-trained model.

One of the simplest lazy classification algorithms is the k-nearest neighbour (k-NN) algorithm [19]. The k-NN algorithm is illustrated in Figure 5.2. In this figure, a triangle represents a touch event that is from the legitimate user "me" and a square represents an event that is from an imposter "not me". A new instance is represented as a question mark in the figure. Depending on the value of k, which is user defined, the new instance's neighbours vote on its class. The nearest neighbours are determined using some form of distance metric, such as Euclidean distance for continuous data, or Hamming distance for binary data. Once the new instance has been classified, it is added to the set of training instances. This set may be restricted in size, or a sliding window can be used to keep the number of training instances small. In this case, old instances will be replaced by new instances (first in, first out). Therefore, once the new data instance has been added to the dataset, the oldest pattern is removed from the dataset. A new unseen data instance is added to the set of training instances, and the next unseen data instance is classified using the new training instance.

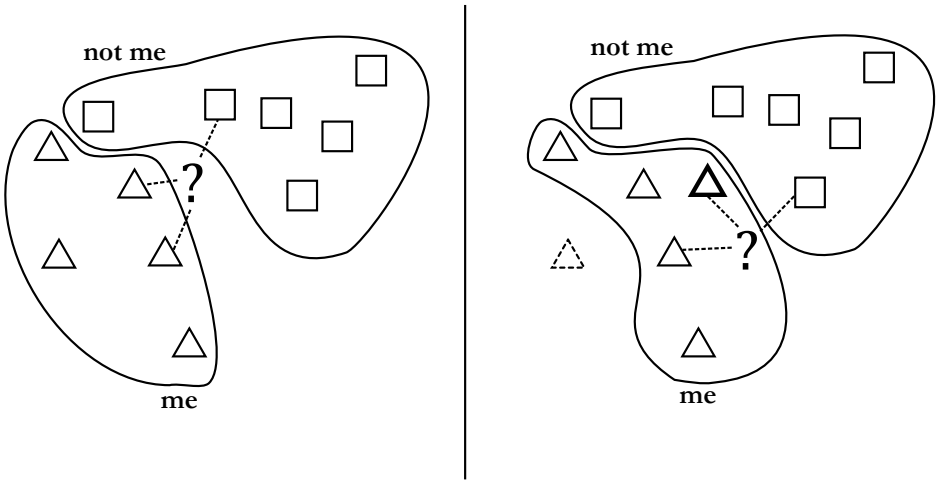


Figure 5.2: k-NN algorithm where $k = 3$, the new classification is added to the set of training instances, and the oldest pattern is removed.

The K* algorithm is based on the idea that entropy is a more reliable

way of measuring the distance between two instances. The entropic distance is based on the Kolmogorov complexity. It is a measure of the number of resources needed to describe an object in some universal description language [51].

The Kolmogorov complexity of a string is the minimal representation describing that string. For instance, the string "aiaiaiai" can be described as 4(ai), which has a length of 5. However, the string "aiciaici" can be described as 2(aici), which has a length of 7. Therefore, it is more complex, because the length of the shortest string that could represent "aiciaici" is longer than the shortest string that could represent "aiaiaiai".

The conditional Kolmogorov complexity is the relative Kolmogorov complexity between two instances [58]. Theoretically, it is the shortest program that can output a given string from another string. Conditional Kolmogorov complexities have been used as a distance measure and as a way to calculate entropy [35, 44].

However, using a single Kolmogorov complexity could bias this measurement, so the probability of one instance arriving at another by doing a random walk is used to approximate the average Kolmogorov complexity [17]. The K* distance function is defined as [17]:

$$K^*(b|a) = -\log_2 P^*(b|a) \tag{5.3}$$

Where:
$K^*(b|a)$ is the entropic distance between instance a and instance b, and
$P^*(b|a)$ is the probability of all paths from instance a to instance b.

The K* algorithm does not depend on a user specified value for the number of neighbours, but does need the blending parameter B. B produces a "sphere of influence" that an instance's neighbours have on it. $B = 0\%$ uses only the nearest neighbour for an attribute, while $B = 100\%$ gives an equal weighting to all training instances. Cleary and Trigg's research shows that a value of $B = 20\%$ works well in most cases, but parameter optimisation shows slight improvement [17].

The classification is performed by calculating the sum of the probabilities between the new instance and all instances in that class. Formally, it is defined as [17]:

$$P^*(C|a) = \sum_{binC} P^*(b|a) \tag{5.4}$$

Where:
$P^*(C|a)$ is the probability of instance a belonging in class C, and

$P^*(b|a)$ is the probability of all paths from instance a to instance b, which is an instance in C.

The class of the new instance is determined as the class with the highest probability.

Because the K* classification algorithm is a lazy classification algorithm, it can be used in the touch biometric system without needing a re-training step in the matcher module. It is therefore a good alternative to using the C4.5 tree algorithm.

5.3 Algorithm Setup

The Weka toolkit was used to provide implementations of the classification algorithms as discussed in Sections 5.1 and 5.2. This section gives an overview of the setup used to conduct the experiment to generate the accuracy results.

5.3.1 Parameter Setup

For the C4.5 tree, the confidence factor c was set to 0.25 and the minimum number of objects m was 3 as these are the defaults set in WEKA and showed acceptable accuracy rates. For the K* algorithm, the recommended value of 20% was used for the blending parameter as discussed in Section 5.2.

The accuracy rates could be improved by using parameter tuning. Parameter tuning tests different combinations of algorithm parameters to achieve the best possible outcome from classification.

However, in the proposed touch biometric system, the format of the touch data is unpredictable and it is important that the system is fast and responsive. Therefore, this step would have to be repeated whenever the system receives new training data. Parameter tuning is not done here, and the results discussed in the next chapter are not optimal. However, the results still give an indication of the accuracy of the system although they may not represent the best possible performance. Future work could focus on finding better accuracy rates for specific implementations of the matcher module.

5.3.2 Threshold Values

The threshold value determines for which application a system could be used. If the threshold is set to be more strict, the system would be used for a high security system. However, if the threshold value is set to be less strict, more

false positives would occur. More false positives would be more useful on a system where high security is not required and usability is a higher priority.

Predicted class outputs for both the C4.5 tree and K* algorithm have a probability estimate associated with them. These probabilities are used to determine whether an input pattern's classification will fall below or above the classification threshold for each class. Values reported in the next chapter for FAR, FRR and the correct percentage are for the default threshold value on predicted probabilities in the Weka toolkit. This value is 0.5.

The ROC curve and AUC give more information that is threshold independent. These measurements give an impression of the overall accuracy of a system, regardless of its threshold value. Whether the threshold is set high or low, a system with a high AUC will be applicable to both situations. The ROC curve and AUC are therefore the most important measurements reported in the next chapter.

The next section describes the definition of "acceptable accuracy" used in this study.

5.4 The Acceptability of Accuracy Rates

Accuracy rates cannot be classified as "good" or "bad" in a global sense – they can only be seen in relation to the intended use of a system. Therefore, the defined accuracy ranges are only set up to guide the discussion of accuracy in the following chapter.

The level at which an accuracy rate becomes "acceptable" is subjective and depends on the application of the system. Therefore, an EER of < 0.2 and an AUC > 0.9 is seen as "acceptable" for the purposes of this discussion as explained below. The following classifications will be used in the rest of this dissertation.

An error rate, that is the EER, FAR or FRR, below 0.5 is better than randomly choosing classifications, but is *not acceptable* as part of the proposed system. Such an error rate would indicate that an attacker would be blocked only about 50% of the time, and legitimate users will be blocked 50% of the time while they are working.

An error rate below 0.2 is seen as *acceptable* in this study. This indicates that attackers will be blocked 80% of the time or more, and that users will be unnecessarily disturbed less than 20% of the time. This is acceptable here because this study simply investigates the distinctiveness of touch biometrics and the results have not been optimised by parameter optimisation or varying threshold values. Furthermore, the proposed touch biometric system would serve as an extra level of security, and is not expected to stand alone as the

only authentication mechanism.

An error rate below 0.05 is seen as *excellent* in this study. A rate lower than 5% shows that only 1 in 20 attackers would be able to circumvent the system on average.

The AUC should be above 0.5 in order for the classification to be seen as *better than worthless*, as defined in Section 2.3.2. However, in this study a value of 0.9 or higher is seen as *acceptable* since it approaches 1 and indicates a good ROC curve.

5.5 Conclusion

This chapter described the classification algorithms used to classify data patterns in the touch biometric system's matcher module. Both an eager classifier and lazy classifier were used for determining the accuracy of touch biometric data. The accuracy of the results is discussed in detail in the next chapter.

Chapter 6

Results

The proposed system consists of four modules: the sensor, feature extractor, matcher and database. Previous chapters addressed these modules and their implementations. This chapter discusses results obtained when testing the matcher module with the datasets described in Chapter 4, using the two classification algorithms discussed in Chapter 5.

The accuracy is analysed to determine how distinct the collected touch biometric data is for different users. The two algorithms used were the C4.5 decision tree algorithm and the K* lazy classification algorithm. The goal of the analysis in this chapter is not to compare the two algorithms, but rather to gain an understanding of how well a real-world implementation of the proposed system would be able to perform. In other words, to determine whether the proposed system is accurate in the general case and not to determine which classification algorithm is most accurate.

The experiment was designed to answer the following questions.

Which touch biometric features offer the most information gain?

Calculating the information gain of recorded features gives insights into which features measured in this experiment were the most influential when making classification decisions. Section 6.1 gives an overview of which features are most informative for classification in the datasets. Information gain and entropy are discussed in Section 5.1.

Can the system be used for identification and for verification?

Section 6.2 provides an overview of the accuracy achieved by both classification algorithms during identification and verification. In this section, the accuracy of the raw data against the accuracy of the gesture data is also

discussed. The accuracy of the raw data against the gesture data provides additional insights into the informativeness of different features across the raw and gesture datasets.

How accurately can touch biometric data be classified? How does the accuracy affect security and usability?

The security of the touch biometric system is dependent on a low FAR. The usability of the system is dependent on a low FRR, meaning that users will not be interrupted by the authentication failover module frequently. The accuracy rates achieved are high, and this indicates that the security does not suffer in favour of the usability. Section 6.3 provides more detailed information about the ROC and the area under the ROC curve values, and how these values determine how well the system performs in terms of its accuracy on various threshold values. It shows that touch biometrics perform well overall and the proposed system could therefore be both secure and usable.

Is the touch data of some users inherently more distinct than the touch data of others?

Section 6.4 investigates whether some users are easier to recognise. If some users are more recognisable than others, the system will be more effective for some users than for others. Therefore, threshold settings should be adjustable per user. Section 6.4 also provides possible reasons for better classification accuracy on some users, including that some users are not accustomed to using an Android smartphone.

Does more training data lead to better accuracy?

Some users may be more recognisable than others because they generated more touch data for training than others. The effect of the amount of training data on the accuracy is investigated in Section 6.4.1.

6.1 Analysis of the Information Gain of Touch Biometric Features

The informativeness of features is analysed here in terms of information gain in the identification and the verification datasets.

6.1.1 Identification Dataset

To determine which were the most informative features in the touch data overall, the raw and gesture datasets' features were analysed according to their information gain values with respect to the user's classification. The results are shown in Tables 6.1 and 6.2.

Feature		**Information gain**
f_b	Touch minor	0.78822
f_a	Touch major	0.78822
p	Pressure	0.66031
x	x-coordinate	0.56714
y	y-coordinate	0.49512
f_θ	Tool orientation	0.22725
f_c	Tool major	0.22484
f_d	Tool minor	0.22484
s_θ	Screen orientation	0.22212
s	Size	0.18535
t	Total time	0.147
i	Pointer ID	0.04869
g	Type of gesture	0.03705
a	Action type	0.00601

Table 6.1: Information gain of the raw features in the identification dataset with respect to the 30 user classes.

The features with the highest information gain for the raw features were the touch minor f_b and major f_a axes. Both of the touch ellipse axes were measured as the same value for the HTC Desire running Cyanogenmod 7, used in this experiment. Other devices may have different results since if they measure distinct values for both. These features describe the shape of the touch area. The pressure applied p also has a high information gain value. The x-coordinate is more informative than the y-coordinate. This could indicate that the horizontal position of touches vary more between different users than the vertical space they use.

The action type a and gesture type g were the least informative in the raw dataset. The information from a could be contained in the total time t. A shorter time indicates a shorter action such as a tap while a longer time indicates a longer action such as a move. The low information gain value for g may be a result of the unprocessed nature of the data – the gesture is not yet complete and therefore the gesture type is not informative.

Feature		Information gain
$i_{gesture}$	Pointer ID	0.8227
$\overline{p}$	Pressure mean	0.4219
$\overline{f_b}$	Touch minor mean	0.3963
$\overline{f_a}$	Touch major mean	0.3963
$\overline{p}$	Pressure standard deviation	0.3156
f_{b_σ}	Touch minor standard deviation	0.302
f_{a_σ}	Touch major standard deviation	0.302
$\overline{y}$	Y-coordinate mean	0.2533
$\overline{f_d}$	Tool minor mean	0.2387
$\overline{f_c}$	Tool major mean	0.2387
$\overline{s}$	Size mean	0.2379
y_e	End y-coordinate	0.2195
y_s	Start y-coordinate	0.2165
$\overline{f_\theta}$	Tool orientation mean	0.2063
$s_{\theta_{end}}$	Screen orientation	0.2045
t_t	Total time	0.179
s_σ	Size standard deviation	0.1675
x_s	Start x-coordinate	0.1453
$\overline{x}$	X-coordinate mean	0.1437
f_{c_σ}	Tool major standard deviation	0.1389
f_{d_σ}	Tool minor standard deviation	0.1389
$\|m\|$	Vector length	0.1137
x_e	End x-coordinate	0.1107
y_σ	Y-coordinate standard deviation	0.1081
x_σ	X-coordinate standard deviation	0.1076
$\overline{m_a}$	Average acceleration	0.0767
g_{end}	Gesture type	0.0645
$\overline{m_v}$	Average speed	0.0611
t_i	Inter-stroke time	0.0602
m_d	Vector direction	0.0417
f_{θ_σ}	Tool orientation standard deviation	0
m_θ	Vector angle	0

Table 6.2: Information gain of the gesture features in the identification dataset with respect to the 30 user classes.

Table 6.2 shows that the most informative feature in the gesture dataset is the pointer ID $i_{gesture}$. As discussed in Section 4.3, a pointer ID higher than 0 indicates that the user used a multi-touch gesture. If only a few participants used multi-touch gestures, this would be an important feature on which to split the data for identification. However, if the sample size is larger with more users utilising multi-touch gestures, the pointer ID may not be as distinguishing. In contrast to the gesture dataset, the pointer ID in the raw dataset had a very low information gain value. Possibly this is because the raw events are not classified into continuous gestures, so multi-touch gestures may introduce noise to the dataset, since they are not linked to other events of the same pointer.

In the gesture dataset the coordinates of the gestures had relatively high information gain values. They are therefore reasonably informative, meaning that where a user touches is also distinct between users. Some participants in the experiment scrolled on the left hand side, and others scrolled on the right hand side. Similarly, some participants in the experiment used a horizontal scroll in either the upper or the lower area of the screen. In the gesture dataset, the mean y-coordinate values $\overline{y}$ were more informative than the x-coordinates $\overline{x}$. Therefore, when an entire gesture is considered, it is important where users touch vertically. Some users have longer scrolling actions than others. Some participants in the experiment scrolled slowly through an article, while others read a full page and then quickly scrolled down to the next page.

The tool orientation standard deviation f_{θ_σ} was always 0 for one user on the HTC Desire used in this experiment. It did not vary during one session of one user's interaction. When analysed closely, it became clear that this behaviour was related to the screen orientation. This value only changed when the screen was rotated from portrait to landscape. This was therefore a value that was calculated by the OS and did not truly represent the tool orientation as described in Section 4.3. The value was duplicating the information from the screen orientation feature s_θ.

The vector-related features in the gesture dataset had low information gain overall. Most of the information contained within the vector was repeated information in a new form and did not give any new information to the algorithms. For example, the vector angle provided no new information and did not aid in classification.

The gesture type g_{end} had low information gain – a single tapping gesture was recorded most often. In the raw dataset, the gesture type was often unknown, since some motion events are recorded before the end of the gesture.

The gesture dataset contained a lot of information that was repeated in another form. For instance, if the phone is held in a landscape orienta-

tion, all the coordinate related features will change significantly. Therefore, screen orientation $s_{\theta_{end}}$ may appear to offer little information gain, but this is perhaps due to the multiple features reporting the same anomaly. The coordinates change completely, already indicating that the screen had been rotated.

6.1.2 Verification Dataset

Figures 6.1 and 6.2 investigate whether the same features are always important for all users. The verification datasets were used to calculate information gain values for each user with respect to the "me" vs. "not me" classes. Each dataset for each user "votes" for the feature with the highest information gain. Therefore, if a feature has a vote count of 30, each file that was generated for that user voted for the same feature as the most informative. For example, all files generated for the user with ID 10 voted for the x-coordinate as the most informative, but only three files generated for the user with ID 5 voted that the x-coordinate was the most informative.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Total
Gesture Type	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Action Type	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pointer ID	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Total Time	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21	0	0	0	0	0	0	0	0	0	21
X-Coordinate	4	0	0	17	3	0	0	0	0	30	0	30	0	0	0	0	30	0	24	0	2	0	0	0	0	0	0	0	28	30	198
Y-Coordinate	0	0	0	12	27	0	0	0	0	0	30	0	0	0	0	0	0	0	6	9	7	0	30	0	0	0	0	0	2	0	123
Pressure	0	0	5	1	0	0	0	7	0	0	0	0	28	8	0	4	0	23	0	6	0	27	0	29	30	0	4	29	0	0	201
Touch Area Size	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Touch Major Axis	21	0	12	0	0	0	10	2	0	0	0	0	0	0	0	26	0	0	0	14	0	3	0	0	0	30	1	0	0	0	119
Touch Minor Axis	0	22	13	0	0	0	20	21	0	0	0	0	2	22	30	0	0	7	0	1	0	0	0	1	0	0	25	1	0	0	165
Tool Major Axis	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
Tool Minor Axis	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
Tool Orientation	0	0	0	0	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30
Screen Orientation	0	0	0	0	0	0	0	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30

Figure 6.1: Information gain votes for the raw features in the verification dataset. Red indicates that no votes were received and green indicates that all files voted for this feature. The remaining values are represented by a colour scaled between red and green.

From these values it is clear that not all users are classified using the same features. For example, while the most files voted for the pressure p as the most informative feature (seen by the highest value for the pressure feature in the Total column), it is only seen as the most informative by one file for the user with ID 4. For this user, the x-coordinate was voted as the most informative 17 times and the y-coordinate 12 times. That is, pressure was less important than coordinates to verify the identity of this user.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Total
Gesture Type	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pointer ID	30	0	19	30	30	0	25	30	0	30	0	30	0	28	30	0	30	30	30	30	10	30	15	16	1	18	0	30	29	0	551
Total Time	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	6	0	6	0	0	0	0	0	0	0	14
Inter-stroke Time	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Start X-Coordinate	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Start Y-Coordinate	0	0	0	0	0	17	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22
End X-Coordinate	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
End Y-Coordinate	0	0	0	0	0	5	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
X-Coordinate Mean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X-Coordinate Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y-Coordinate Mean	0	0	0	0	0	6	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10
Y-Coordinate Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	2
Pressure Mean	0	0	1	0	0	0	4	0	0	0	3	0	22	0	0	1	0	0	0	0	0	0	0	0	5	0	18	0	0	30	84
Pressure Standard Deviation	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	2	6	0	7	0	0	0	0	43
Touch Area Size Mean	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	12
Touch Area Size Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	3
Touch Major Mean	0	11	0	0	0	1	0	0	0	5	0	3	0	0	0	0	0	0	0	0	0	0	0	0	3	0	4	0	0	0	27
Touch Major Axis Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	4	0	0	0	0	0	0	1	3	0	2	0	0	0	0	12
Touch Minor Mean	0	8	0	0	0	0	0	0	0	11	0	2	0	0	0	0	0	0	0	0	0	0	0	0	21	0	8	0	0	0	50
Touch Minor Axis Standard Deviation	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	6	0	0	0	0	0	0	5	5	0	3	0	0	0	0	21
Tool Major Axis Mean	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	6
Tool Major Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tool Minor Axis Mean	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3
Tool Minor Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tool Orientation Mean	0	0	0	0	0	2	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21
Tool Orientation Standard Deviation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Screen Orientation	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11
Vector Angle	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Vector Direction	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Vector Length	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
Average Speed	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Average Acceleration	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6.2: Information gain votes for the gesture features in the verification dataset. Red indicates that no votes were received and green indicates that all files voted for this feature. The remaining values are represented by a colour scaled between red and green.

It is interesting to note that there is more variety in the most important features in the raw dataset compared to the gesture dataset. That is, many users have different important features in the raw dataset, but most users' files voted for the pointer ID $i_{gesture}$ feature in the gesture dataset. This indicates that most of the trees generated by the C4.5 algorithm based on the gesture dataset would first split on pointer ID, before using the other features as secondary discriminators.

Figures 6.1 and 6.2 show that some features are overall important for the classification of many users. The values reflect a similar order to the identification dataset as shown in Tables 6.1 and 6.2.

6.1.3 High vs. Low Information Gain

When both the raw and gesture datasets are considered, the information gain values show that features related to the physical shape of the finger – the pressure applied to the screen (p and $\overline{p}$) and the touch ellipse's axes (f_a, f_b, $\overline{f_a}$ and $\overline{f_b}$) were the most informative features.

Derived values in the gesture dataset such as vector direction m_d and vector angle m_θ are the least informative.

To investigate the differences between the highest and lowest information gain two graphs are plotted: the pressure applied by each user and the vector angle of each gesture for each user. These features are chosen to compare the true range of values that were recorded for each feature. Pressure is shown to be informative in most cases, and vector angle is shown to be uninformative in most cases.

The pressure values for all data points in the identification dataset, plotted for different users, is shown in Figure 6.3. This figure plots the different values for pressure for each user. It shows that different users have different pressure ranges. This figure shows clear variances between users for screen pressure.

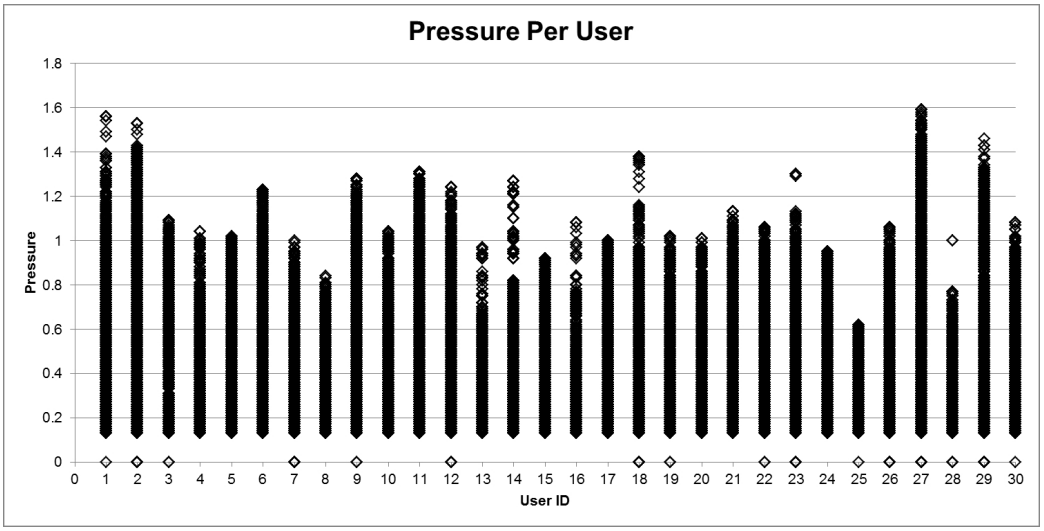


Figure 6.3: Comparison of the pressure rates recorded for each user ID.

The vector angles are plotted in Figure 6.4. In comparison to the pressure applied to the screen per user depicted in Figure 6.3, the vector angle's data points were spread out and do not show a pattern of being distinct per user.

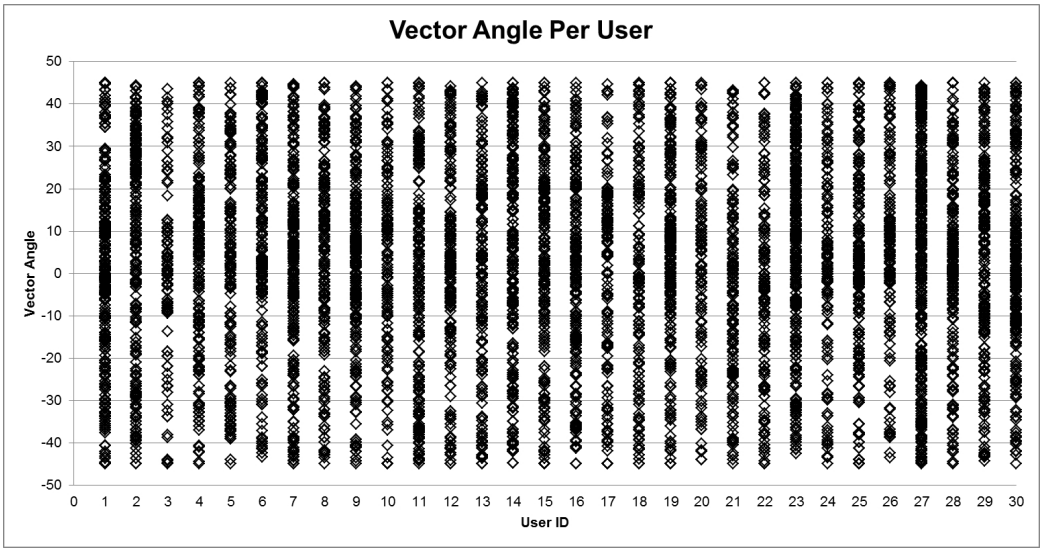


Figure 6.4: Comparison of the vector angles recorded for each user ID.

6.1.4 Conclusion

The pointer ID $i_{gesture}$ is the most important of the gesture features and the pressure p is the most important of the raw features in the verification dataset. The derived gesture features have low information gain – the vector angle m_θ, direction m_d, the average speed $\overline{m_v}$, acceleration $\overline{m_a}$ and length $|m|$.

6.2 Classification Accuracy Rates

This section discusses the accuracy rates that were achieved when classifying users according to their touch biometric data in all datasets. The description of the accuracy measurements reported here are discussed in Section 2.3.

6.2.1 Identification Classification Accuracy

The identification accuracy gives an indication if one user can be identified from the set of all 30 users. The dataset is described in Section 4.4.1 as the dataset of all users. To achieve statistically representative rates, 10-fold cross validation was used on this set. The use of 10-fold cross validation is described in Section 4.4.1. The accuracy of the classification using the C4.5 tree algorithm and the K* algorithm is shown in Tables 6.3 and 6.4.

Measurement	C4.5	K*
Average AUC	0.904 ± 0.039	0.973 ± 0.011
Average EER	0.148 ± 0.072	0.089 ± 0.025
Average FAR	0.008 ± 0.003	0.009 ± 0.005
Average FRR	0.266 ± 0.092	0.303 ± 0.118
Percent correct	76.329%	73.664%

Table 6.3: Accuracy when identifying users based on the raw dataset. FAR, FRR and percent correct are reported for a threshold value of 0.5.

Measurement	C4.5	K*
Average AUC	0.731 ± 0.089	0.858 ± 0.059
Average EER	0.444 ± 0.220	0.205 ± 0.062
Average FAR	0.020 ± 0.006	0.020 ± 0.007
Average FRR	0.614 ± 0.164	0.618 ± 0.186
Percent correct	41.514%	41.948%

Table 6.4: Accuracy when identifying users based on the gesture dataset. FAR, FRR and percent correct are reported for a threshold value of 0.5.

Between the raw and gesture datasets of all users for identification, the raw dataset had better accuracy rates than the gesture dataset. This is perhaps because there is a larger amount of data available on each user and because the features have better information gain than those in the gesture datasets. In the gesture dataset, there are more features for each data instance. However, a lot of information is lost during the feature extraction process, when features are aggregated. In the case where this biometric system would be used for identification as well as verification, the raw dataset could be used without any feature extraction and perform well. However, classification on the raw data requires more time and computing resources.

The AUC value is low and the EER is high for both algorithms in the gesture dataset. A low AUC and high EER shows that the accuracy is low for most threshold values. This indicates that it is not possible to easily balance FAR and FRR of the gesture dataset.

In both datasets, the FAR is acceptable, because only 0.8% to 2% of imposters were on average accepted as legitimate users. However, the FRR is very high in the gesture dataset. This value indicates that there is a lack of knowledge during classification and most users are rejected, but the system

would still be secure at this threshold value. The FAR is not very high, so imposters will not easily be accepted as legitimate users. However, the FRR is so high that the system would be unusable, because most legitimate users will be rejected 60% of the time in the gestures dataset. Note that these values are reported for a threshold value of 0.5 and are indicative of the performance of the system if an average threshold value is chosen. The system could be adjusted for high security applications if the threshold is made stricter. However, the AUC and EER indicate that the gesture dataset will not perform well even if the threshold value is adjusted.

6.2.2 Verification Classification Accuracy

The verification accuracy describes how accurate the system would be as an authentication mechanism as described in Section 2.2.3. It is an indication of how well the system would perform in a real-world situation. This dataset is described in detail in Section 4.4.2. The accuracy of the classification using the C4.5 tree algorithm and the K* algorithm is shown in Tables 6.5 and 6.6.

Measurement	**C4.5**	**K***
Average AUC	0.875 ± 0.034	0.927 ± 0.025
Average EER	0.177 ± 0.041	0.148 ± 0.035
Average FAR	0.227 ± 0.058	0.288 ± 0.077
Average FRR	0.100 ± 0.033	0.062 ± 0.021
Average percent correct	83.720 ± 3.812	82.671 ± 4.298

Table 6.5: Accuracy when verifying users based on the raw datasets. FAR, FRR and percent correct are reported for a threshold value of 0.5.

Measurement	**C4.5**	**K***
Average AUC	0.819 ± 0.072	0.786 ± 0.087
Average EER	0.209 ± 0.086	0.275 ± 0.080
Average FAR	0.254 ± 0.111	0.421 ± 0.140
Average FRR	0.145 ± 0.087	0.170 ± 0.073
Average percent correct	80.584 ± 6.743	71.520 ± 7.957

Table 6.6: Accuracy when verifying users based on the gesture datasets. FAR, FRR and percent correct are reported for a threshold value of 0.5.

In the verification tests, the raw dataset still had higher classification accuracy rates, but the gesture datasets' accuracy improved in relation to the raw datasets. That is, the gesture dataset still did not have higher accuracy than the raw dataset, but the accuracy was not as low as during identification. According to these results, the system generally accepted more imposters than it rejected genuine users for a mid-range threshold. This is the inverse of the situation during identification. During verification, there are only two classes to choose from, so there is a higher chance that a data pattern would be incorrectly classified as one legitimate user.

It is important to note that, although identification is a more difficult classification task than verification, as described in Section 2.2.3, the case is slightly different here. The verification datasets, as described in Section 4.4.2 explicitly excluded any touch data from the attacker. The identification dataset, as described in Section 4.4.1 contained all users and attempts to identify them using new data on previously seen users. Therefore, verification was a more difficult classification task in this experiment.

According to the results given here, the K* algorithm on the raw dataset achieved a high value for AUC overall, and its best AUC was achieved during verification. The same pattern emerges for the EER values. The FAR is very high, for a threshold value of 0.5. However, the EER was lower, indicating that there exists a more optimal compromise between FRR and FAR.

Therefore, for verification, the system can be expected to achieve EERs between 0.15 and 0.2 if the raw dataset is used. This means that if the FAR and FRR are equal, the error would be between 15% and 20%. These values are too high to be used as a high security application, since 20% of imposters could be let into the system. However, at least 80% of attackers would need to breach an extra layer of authentication on a smartphone that would otherwise require no authentication. Therefore, for the intended purpose of the system, the accuracy is acceptable, but should be improved in future work.

Since the FAR, FRR and the percent correct are dependent on the threshold value used, they will not be included in subsequent discussions of the system's accuracy.

6.2.3 Conclusion

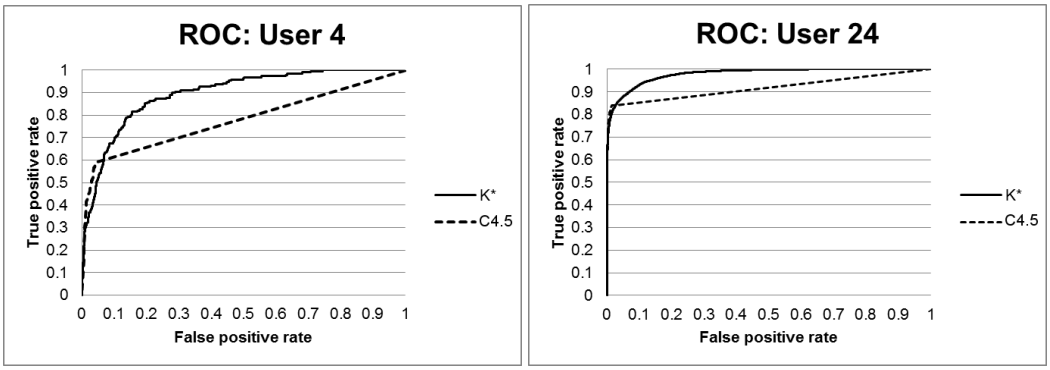
The best area under the ROC curve that could be achieved was 0.973 and the corresponding equal error rate was 0.089. These rates are worse than reported rates from previous work [29, 32, 50]. However, this study was conducted on real-world touch interactions, and the feature extraction process was less specific in terms of which features were used for classification.

6.3 ROC Curves

Receiver Operator Characteristic (ROC) curves give an indication of the accuracy of a system across different threshold values, and the AUC is a single number representing that accuracy. ROC curves are discussed in detail in Section 2.3.2.

According to the AUC value, the K* algorithm generally had better accuracy than the C4.5 tree algorithm. Analysis of the ROC curves shown in Figure 6.5a of a user from the gesture datasets shows that the C4.5 tree algorithm has a less smooth curve between threshold values of 0.5 and 1. The C4.5 tree algorithm has fewer points on the curve since some threshold values result in the same predictions, because of the structured nature of the modelled tree. This indicates that it was more difficult to balance the FAR and FRR of the tree than it was for the K* algorithm. User 4 is a representative of the average AUC for both algorithms in the gesture dataset. The trend was similar for other users.

The same pattern is seen to a lesser extent in the raw data set as shown in Figure 6.5b. Here user 24 was chosen as a representative of the average AUC. Therefore, K* algorithm could more easily be used for low or high security systems using any threshold value. However, as shown in the graph, the C4.5 tree algorithm had lower error rates for lower threshold values for both datasets. The C4.5 tree algorithm is a better *conservative* classifier, meaning it would be used in high security applications. However, the K* algorithm is a more *liberal* classifier, and is more useful in systems where usability is important. The difference between conservative and liberal classifiers is discussed in Section 2.3.2.



(a) Gesture Dataset (b) Raw Dataset

Figure 6.5: The ROC curves of the K* algorithm against the C4.5 tree algorithm for an average user from the gesture and raw datasets during identification.

For the purposes of the proposed touch biometric system, a more conservative classifier is preferred. However, if users are allowed to adjust the threshold value as described in Section 3.2, then the K* algorithm is preferred since it has a smoother curve and has low error rates for all threshold values. Furthermore, the K* algorithm does not need to be retrained periodically and is therefore more suited to the proposed system. However, classifying new touch data instances could be too resource intensive and slow to be practically applicable.

6.4 Per User Accuracy

Some users were classified more or less successfully than others. This section discusses the accuracy per user in the datasets.

As shown in Figures 6.6 and 6.8, the AUC varies between users and is very dependent on the dataset and algorithm used. The same trend is shown inversely for the EER in Figures 6.7 and 6.9.

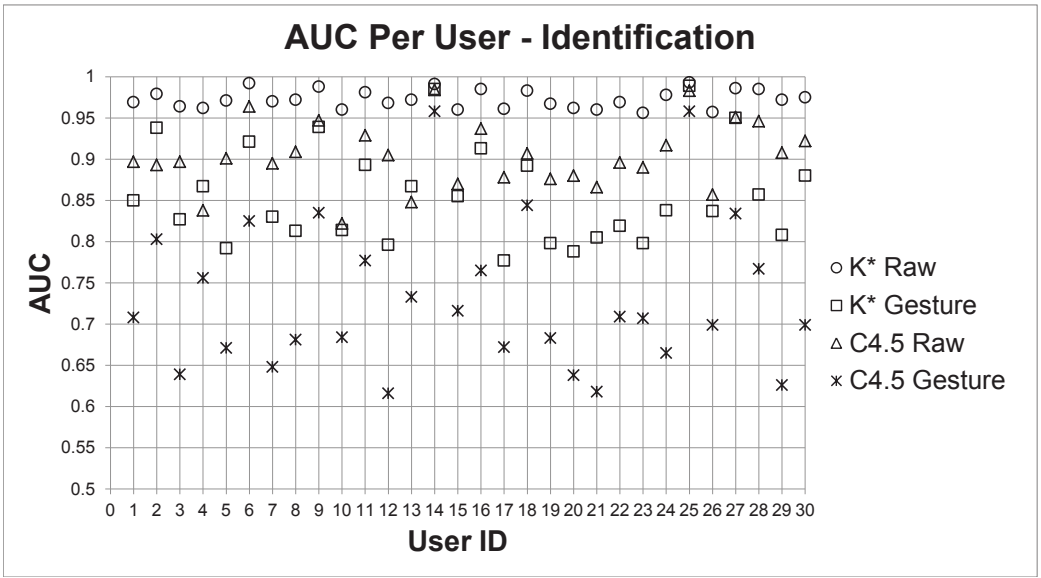


Figure 6.6: Comparison of the average AUCs for each user ID during identification.

Some users could only be classified with good AUC values if the correct algorithm and dataset combination was used. Some algorithms will therefore be more accurate for some users. Users such as 7, 8, 12 and 21 had very low AUC values during identification using the C4.5 tree algorithm on the gesture

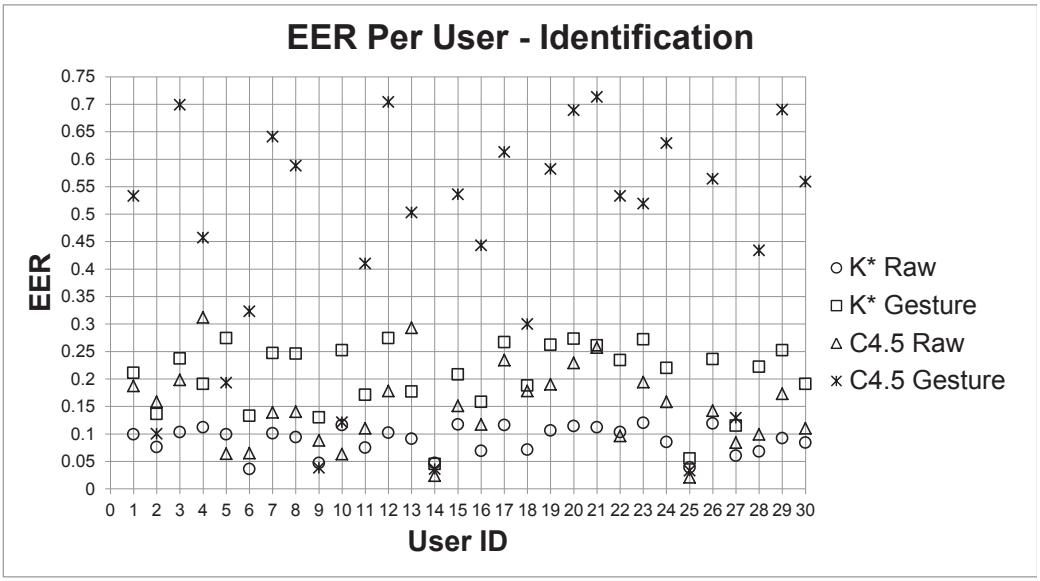


Figure 6.7: Comparison of the average EERs for each user ID during identification.

dataset, but the classification could still result in good accuracy if the K* algorithm was used on the raw dataset.

It can also be seen from Figures 6.6 to 6.9 that some users were generally more recognisable than others, since they had better accuracy rates than other users with any algorithm and dataset combination. The classification algorithms achieved higher AUC values for both datasets and both algorithms. For instance, users 6, 9, 18, 25 and 27 were classified with high AUC values in most situations.

Users 6 and 9 turned the phone around to landscape when browsing the Internet. This is a distinct behaviour and may have resulted in the high accuracy during classification. The information gain votes of these two users' datasets in Figures 6.1 and 6.2 show that screen orientation and tool orientation are important features that arose from this behaviour.

User 14 was observed to have very long nails, and complained that the nails interfered with typing on the touchscreen. This resulted in distinct pressure measurements as shown in Figures 6.1 and 6.2.

Users 18 and 27 were not used to using an Android smartphone, and had limited experience using a touchscreen. These users took longer to perform gestures, or performed gestures slowly and had to repeat them. For example, some participants were not familiar with the pattern unlock mechanism. They attempted to unlock the screen more often than other participants did.

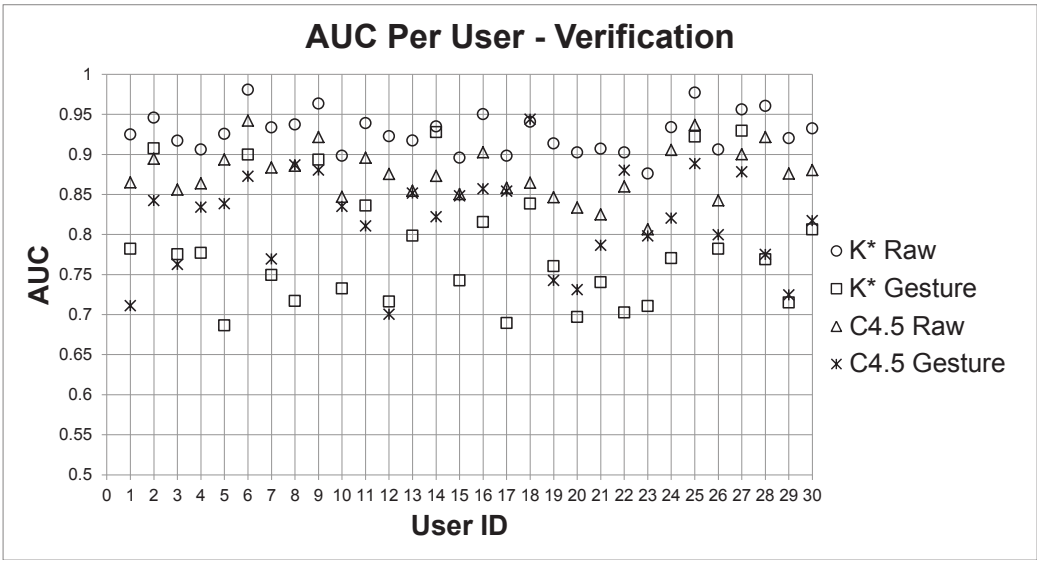


Figure 6.8: Comparison of the average AUCs for each user ID during verification.

Their interactions may be more distinct than users who are familiar with Android smartphones.

User 25 disclosed a touch affecting disability in the experiment questionnaire. The expected system behaviour is that this user will have a very distinct touch pattern. The graph shows that user 25's data results in high accuracy rates across all algorithm and dataset combinations. Figures 6.1 and 6.2 show high information gain from the pressure and touch ellipse features.

These observations during the experiment show that each user has distinct behavioural patterns. Experienced smartphone users may have developed a set of unique habits, such as turning the phone landscape. Attackers who are not used to a user's phone may not exhibit the same combination of behaviours as the phone's owner.

For some users, the high classification accuracy may only be as a result of generating more touch data by repeating actions or by simply taking longer to complete the tasks. The addition of more training data may lead to better results. The effect of the amount of training data and the EER is investigated in the next section.

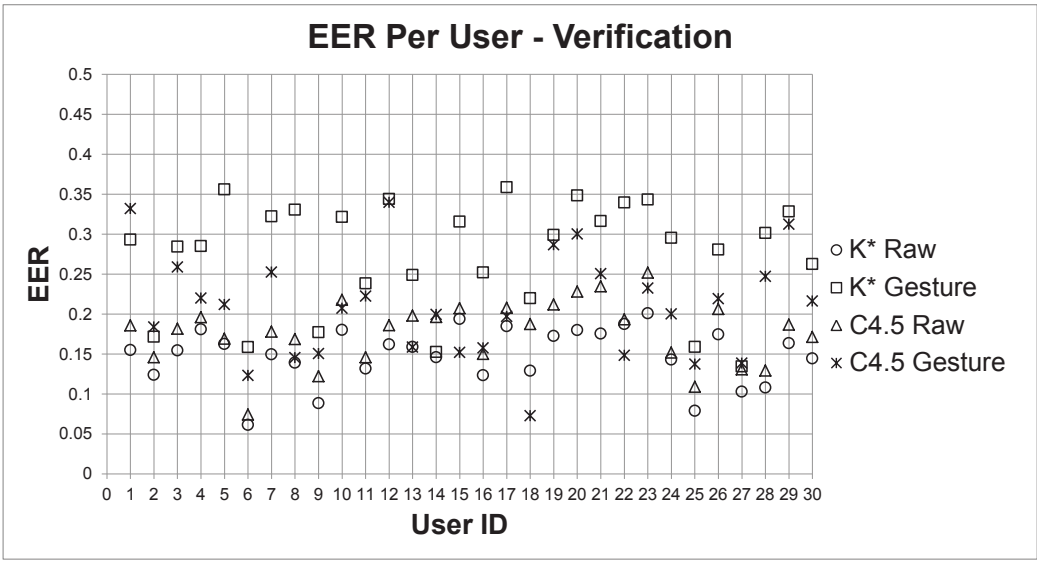


Figure 6.9: Comparison of the average EERs for each user ID during verification.

6.4.1 Effect of the Amount of Touch Data on Accuracy and Performance

In the experiment the participants were limited to a time period of 20 minutes. Some users completed the 6 tasks within 10 minutes. Therefore, the amount of data collected from each user was not large in relation to what a real-world system would collect over time from a smartphone's user.

It is suggested that the system will improve over time as the amount of collected data is increased. That is, if the system is used for a long period the FAR and FRR will decrease and the overall security and usability will improve.

Classification was more accurate for some users in the datasets. It is possible that these users performed more or longer gestures, resulting in more data for these users. To investigate this possibility, the datasets were reduced to be no bigger than the size of the smallest dataset – the user that generated the least touch data.

Figure 6.10 shows the result of reducing all datasets to the same size on the raw verification dataset, using the K* classification algorithm. As can be seen, the EER increased for most users.

Figure 6.11 shows the same process with the C4.5 tree algorithm resulted in similar increases in EER, except for two user datasets that were classified with fewer errors. This may indicate that overfitting occurred when the

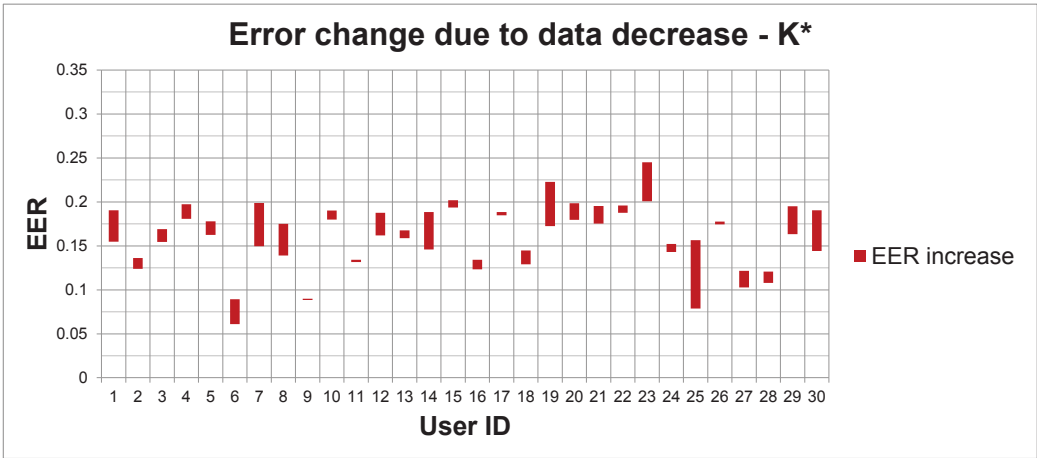


Figure 6.10: Change in EER when all datasets are reduced to the same size.

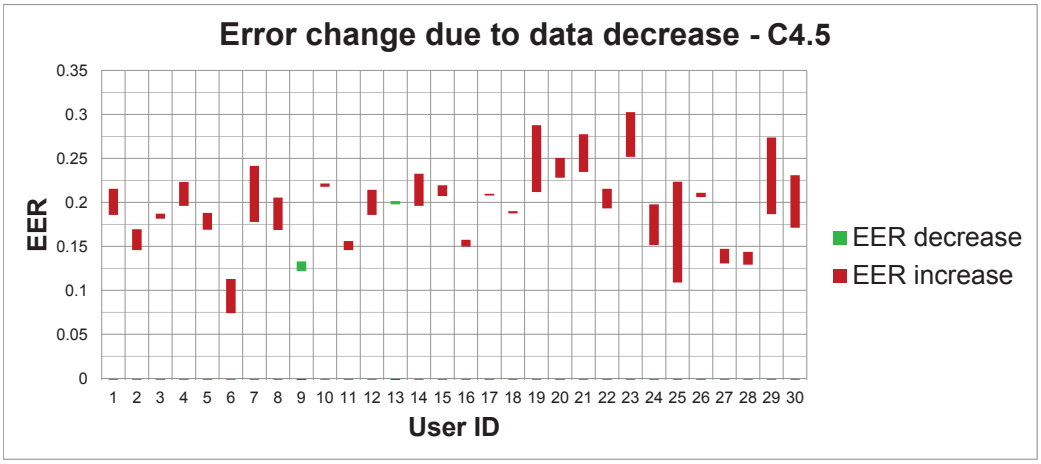


Figure 6.11: Change in EER when all datasets are reduced to the same size.

classifications were modelled. However, the values are small and do not indicate that this trend would occur reliably in other tests.

Figures 6.10 and 6.11 show that decreasing the amount of data available increases errors in the general case. However, some user datasets always achieve high accuracy, and decreasing the amount of data did not level out error rates across users.

Computational complexity of large datasets

It is desirable to use a lazy classification algorithm like the K* algorithm since it is easy to add new training data instances to the database module.

However, classification is slower since the entire set of training instances need to be compared to the new instance before a classification decision is made. This process is usually $O(n)$ time complexity for each new data instance. That is, each training data instance inside the training data set needs to be compared individually to a new data instance to classify that new data instance. Therefore, the system should use datasets that are small enough to execute in a reasonable amount of time, but still large enough for high accuracy.

If a large dataset is required, eager classification algorithms, such as the C4.5 tree algorithm, should be considered first. These algorithms only need to reference a pre-built model that was built on a set of training data. Therefore, the processing time will be less than $O(n)$ since in the worst case all the nodes in the tree need to be compared to a new data instance, and this will still be less than the total number of training instances. However, eager classification algorithms are less adaptable to new information. Whenever a new data instance is added to the training set, the model needs to be rebuilt. Therefore, depending on the algorithm used to build the model and the number of new training instances being added to the database, eager algorithms could perform worse than a lazy algorithm.

The K* algorithm uses a sliding window to limit the number of instances in the system's database module. When generating these results, the algorithm's sliding window was not used, because of the small amount of data available. However, a real-world system should limit the size of the database by using the sliding window, to reduce the time required to classify new instances.

It is not known whether new instances in the database will lead to worse accuracy if those instances vary greatly from the existing training instances. This could occur if the user has changed their behaviour over a period of time. Future work should investigate the effect of real-world use and the changing behaviour of users over time.

6.5 Conclusion

The experiment was conducted to answer the questions as outlined in the beginning of this chapter. They are reviewed here.

Which touch biometric features offer the most information gain?

Pressure and other metrics related to the size and shape of the users' fingers are the most informative. The raw dataset contained more informative

features than the gesture dataset. This is because the gesture dataset aggregated information, diluting the informativeness of features. Between the raw and gesture datasets, the raw dataset had better accuracy overall. This indicates that there is no need for significant pre-processing of touch biometric data to achieve acceptable error rates for this system.

Can the system be used for identification and for verification?

When the raw datasets were used, the system achieved acceptable accuracy in both tasks. However, the system performed better during identification since all users were known to the system. This does not represent a real-world implementation, because an attacker will not be known to the authentication system. For a threshold value of 0.5, the FRR was more acceptable during verification than it was during identification.

How accurately can touch biometric data be classified? How does the accuracy affect security and usability?

The EER and AUC values show that if the raw dataset is used, both security and usability can be achieved simultaneously without sacrificing one for the other.

Is the touch data of some users inherently more distinct than the touch data of others?

Some users were generally more recognisable than others. This indicates that touch biometrics are unique to each user and are distinct so that some users are very recognisable. Unique behaviours exhibited by participants and observed during the experiment translated into high accuracy rates. Some behaviours were related to the level of experience of each participant with smartphones. This indicates that attackers may be easy to recognise because they may not have experience with a user's particular phone model and version of Android. However, it also indicates that attackers could potentially train themselves to mimic users' behaviours.

Does more training data lead to better accuracy?

More data in most cases led to better accuracy. However, if the system has too much data, the algorithm may struggle to classify all the data and the phone's performance may suffer because of the computational load of classification. This is mostly applicable to a lazy classification algorithm such as the K* algorithm. Eager algorithms such as the C4.5 tree algorithm

only build a model periodically and then use that model for classification. Therefore, the performance of the C4.5 tree algorithm will remain constant even if a large amount of data needs to be classified. To reduce the amount of training data, a sliding window of new data should be used. This reduces the space and processing requirements of the algorithm, and also keeps the system from remembering old user behaviours.

The next chapter concludes this dissertation by summarising the findings and contributions, as well as proposing future work to be done.

Chapter 7

Conclusion

This study investigated how to improve authentication on smartphones in terms of both security and usability. Smartphones need to be well-protected with effective, secure and usable authentication systems, to keep the information stored on smartphones confidential, accessible and intact. It was suggested in this dissertation that one way to implement better authentication on smartphones was through the use of touch biometrics – biometric features measured using touchscreens on smartphones. Furthermore, it was suggested that these features could be measured continuously, enabling authentication at all times and creating a constant barrier between the smartphone and attackers.

Therefore, this study was conducted to determine how authentication can be done more easily and continuously using touch-based biometrics measured by a touchscreen on an Android smartphone. The study aimed to find a model for touch biometrics, enabling the easy implementation of touch biometrics for authentication on a variety of smartphone models and versions of Android. This was done by analysing previous work in the field of touch biometrics and implementing the findings of other authors. The model's practical implementation was tested using an explorative experiment to find challenges that will be faced if touch biometrics are implemented in a real-world authentication system. The analysis of the experiment results, in terms of the accuracy rates, showed that touch biometrics could successfully determine and verify the identity of 30 experiment participants.

7.1 Summary of Findings

An in-depth analysis of previous work informed the study conducted in this dissertation and provided a foundation for the formulation of a model for

continuous touch biometrics on smartphones.

The proposed model was designed to address the limitations of previous research, such as limiting the actions of the users, not fully discussing results or datasets and the exclusion of multi-touch gestures. The model formalises the requirements for an implementable and extensible continuous touch biometric authentication system.

This model was tested by conducting an experiment which simulated the various parts of the model. The accuracy of the classification of touch biometric data was tested by collecting this data using an application on an Android smartphone.

Initial investigations into the implementation of a touch data collection tool on an Android smartphone revealed that such an application would require specific system-level permissions. Therefore, it was suggested that any implementation of a touch biometric system be incorporated into the operating system, both to avoid the limitations placed on non-system-level applications and to improve the security of the authentication system itself by making it more difficult to remove from the smartphone. The experiment conducted in this study was successful in incorporating the authentication system into the Android operating system. Therefore, the study has shown that such an implementation is possible.

The system-level data collection tool allowed the experiment in this study to investigate any action that could be performed by a user while using any application on the phone. This study was therefore not limited by specific actions that users could perform, and captured real-world continuous touch interactions with a smartphone. Furthermore, multi-touch gestures were also recorded. Therefore, the recorded data was complete in its raw form.

Once the data was collected, the accuracy of the system was analysed according to biometric accuracy rates of two classification algorithms – the K* algorithm and the C4.5 decision tree. This analysis showed that good accuracy rates could be achieved on the touch data. The best area under the ROC curve that could be achieved was 0.973 and the corresponding equal error rate was 0.089. This accuracy was achieved using the set of raw data and classified with the K* classification algorithm. This accuracy was achieved on the identification dataset which contained more information for each user than the verification dataset. The best accuracy on the verification dataset was also achieved using the K* classification algorithm, which was an area under the ROC curve of 0.927 and a corresponding equal error rate of 0.148. Therefore, when the system is balanced in terms of the level of strictness of the threshold value, 15% of attackers will be allowed access, and 15% of legitimate users will be blocked.

A balanced application of this algorithm would not be secure or usable

enough for certain applications. However, the threshold value could be used to customise the level of security against usability required for a specific application. The threshold value could be changed to improve either the false accept rate or false reject rate, depending on the intended application of the system. Furthermore, the proposed solution describes a failover approach which prompts the user for a password if the system locks the user out. This ensures that the security of a system will not depend solely on the touch biometric authentication mechanism.

The analysis of the touch data showed that the pressure touch feature in the raw dataset and the pointer ID in the gesture dataset were the most informative in terms of information gain. The derived gesture features such as vector angle and direction had low information gain. This finding supports those of Frank et al. [32] and Saevanee and Bhatarakosol [68], but contradicts the findings of Li et al. [50]. Frank et al. found that the mid-stroke area covered, the stroke velocity and the mid-stroke pressure offered the highest relative mutual information gain. The work of Saevanee and Bhatarakosol [68] found that pressure was important in classification. Contrastingly, in the work of Li et al. [50] identified pressure-related metrics as poor metrics for all gestures. The remaining metrics such as the starting coordinates, starting moving direction and average touch area were identified as good metrics since the datasets had high amounts of differences between these metrics. These studies used different methods to extract features from raw data on different operating systems and devices.

The differences in the findings show that more investigation is required into the effect of various factors on the touch biometric features. A large-scale study involving different operating systems and different devices is therefore needed.

It was found in the analysis of the touch data, that some users have more distinct touch behaviours and therefore produce higher accuracy rates when classification is performed. High accuracy rates correlated to observed unique behaviours during the experiment. These behaviours may change steadily over time. The proposed continuous touch biometric system ensures that the implementation of a continuous touch biometric system be kept up to date with new usage patterns, and that old patterns should be discarded as new patterns are added.

This study has therefore successfully shown that continuous touch biometrics can be successfully implemented on smartphones and that such a system could be set to be both secure and usable.

7.2 Main Contributions of this Study

The proposed continuous touch biometric system overcomes the limitations of previous research by implementing the following features, as discussed in Section 3.2.

The system has no explicit enrolment phase, since learning is performed while the user performs normal actions on the smartphone.

The system contains a failover module, that still prompts the user for a traditional password. Therefore, the security of the system is not entirely dependent on the touch biometric system.

The system is customisable, since the threshold value can be changed to favour either security or usability for different applications.

The prototype showed that the system can be embedded into the operating system by altering the system level applications that record touch data.

The data collected included multi-touch gestures and thereby ensured that the data included all gestures performed by participants.

Furthermore, the analysis of the datasets used showed that no pre-processing is required to make touch biometric data classifiable.

The results presented in this study reviewed threshold-independent accuracy measurements.

The system is adaptable and independent of the type of device and operating system, since touch databases are kept separate according to the user's environment.

Finally, the system does not depend on any external hardware and the classification algorithm may be run directly on a device.

This study has therefore improved and extended on previous research and has set a foundation for the requirements of subsequent studies on the implementation of a continuous touch biometric system.

7.3 Future Work

Smartphones have many different sensors to facilitate their different functions, such as accelerometers to enable the tilt sensitivity of the screen. The collection of sensors on smartphones represents a good opportunity to combine many biometric features into one authentication system. The combination of various biometric features into one biometric system is referred to as a *multimodal* biometric system. Multimodal biometric systems have been shown to be more accurate and robust against attacks than unimodal biometric systems [41]. Previous research has shown that the use of various

sensors of smartphones for biometric recognition is growing. Research has been done into the use of the accelerometer [23, 24, 40], the camera [2] and cellular network data [46, 55] for authentication. Future work could combine the touch biometric authentication system proposed in this study with more data from all available sensors on the smartphone to create a robust, multimodal biometric system.

The field of touch biometrics is a new field and therefore requires more investigations into the effect of various factors on the touch data. There is no clear agreement in existing research on the distinctiveness of various touch biometric features, or the appropriate feature extraction process that should be followed. Future work should focus on confirming the work of previous research by repeating studies on different devices in a large-scale study.

Future work should focus on implementing a full prototype authentication system to be tested by users for extended periods of time. Only if a prototype is developed will usability concerns become apparent. Similarly, to assess the real security of such a system, more investigations should be conducted into the accuracy when attackers attempt to mimic user behaviours, or if the attackers gain access to the database of touch data.

A specific challenge that would need to be addressed is how to accommodate changing user behaviour due to new environments, activities or social situations. For example, a user may only occasionally use their phone to show photos. When it is used for the first time, the swiping action may not yet have been presented to the authentication system and it may cause the smartphone to lock.

Similarly, the expected behaviour if a user hands their phone to a friend would be that the system should lock. If this happens, there should be some way for the user to unlock the phone and not introduce their friend's touch information into the system.

More advanced or specialised classification algorithms could improve the accuracy of touch biometric systems. Future work could extend known algorithms that are used for biometric classification. For example, the possibility of using a random forest classifier instead of a single decision tree is discussed by Angulo [8]. Furthermore, a new algorithm could be developed specifically to classify biometric data.

Future work should extend investigations to more operating systems and more advanced hardware, including other mobile devices such as tablets. It is possible that accuracy will improve in relation to hardware improvements on mobile devices.

This study and previous work has focused on a relatively small number of participants. Future studies should test a real-world simulation, where the number of participants is large and the environment is less controlled. For

example, a study could be conducted remotely by collecting touch data from willing participants via a downloadable application.

7.4 Conclusion

This dissertation investigated the possibility of using touch biometrics on smartphones to authenticate users continuously. This study indicates a future of more secure and usable authentication on smartphones without requiring users to purchase new hardware and without requiring more effort on the users' part to authenticate. This study has shown that touch biometrics for continuous authentication can be implemented and that the classification of touch biometric features to authenticate has high accuracy rates. Future work will focus on improving the security and usability of this authentication mechanism by improving accuracy rates and on the implementation of a working touch-based authentication system on various devices.

Bibliography

[1] A. Adams and M. A. Sasse. "Users Are Not the Enemy". In: *Communications of the ACM* 42(12) (Dec. 1999), pp. 40–46.

[2] A. Agrawal. "User Authentication Mechanisms on Android". PhD thesis. Indian Institute of Technology, Bombay, 2013.

[3] F. Aloul, S. Zahidi, and W. El-Hajj. "Two factor authentication using mobile phones". In: *IEEE/ACS International Conference on Computer Systems and Applications (AICCSA)*. Rabat, Morocco, May 2009, pp. 641–644.

[4] D. Amitay. *Most Common iPhone Passcodes*. 2011. URL: http://danielamitay.com/blog/2011/6/13/most-common-iphone-passcodes (visited on 02/19/2013).

[5] L. Andrews. "Passwords Reveal Your Personality". In: *Psychology Today* (Jan. 2002), p. 16.

[6] Android Open Source Project. *MotionEvent Documentation*. 2013. URL: http://developer.android.com/reference/android/view/MotionEvent.html.

[7] Android Open Source Project. *WindowManager.LayoutParams Documentation*. 2013. URL: http://developer.android.com/reference/android/view/WindowManager.LayoutParams.html.

[8] J. Angulo. "Usable privacy for digital transactions: Exploring the usability aspects of three privacy enhancing mechanisms". Licentiate thesis. Karlstad University, 2012, pp. 36–37.

[9] A. J. Aviv, B. Sapp, M. Blaze, and J. M. Smith. "Practicality of Accelerometer Side Channels on Smartphones". In: *Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC'12)*. Orlando, Florida: ACM, Dec. 2012, pp. 41–50.

[10] D. Balfanz, G. Durfee, D. K. Smetters, and R. E. Grinter. "In Search of Usable Security: Five Lessons from the Field". In: *Security & Privacy, IEEE* 2(5) (Sept. 2004), pp. 19–24.

[11] J. Bonneau. "The Password Thicket: Technical and Market Failures in Human Authentication on the Web". In: *The Ninth Workshop on the Economics of Information Security*. Cambridge, Massachusetts, USA, 2010.

[12] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano. "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes". In: *2012 IEEE Symposium on Security and Privacy (SP)*. San Francisco, USA: IEEE, May 2012, pp. 553–567.

[13] A. P. Bradley. "The use of the area under the ROC curve in the evaluation of machine learning algorithms". In: *Pattern Recognition* 30(7) (1997), pp. 1145–1159.

[14] L. J. Camp. "Guest Editor's Introduction: Security and Usability". In: *IEEE Technology and Society Magazine* 26(1) (2007), pp. 3,24.

[15] S. Chiasson, A. Forget, R. Biddle, and P. C. van Oorschot. "Influencing Users Towards Better Passwords: Persuasive Cued Click-Points". In: *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction*. Vol. 1. Swinton, UK: British Computer Society, 2008, pp. 121–130.

[16] N. L. Clarke and S. M. Furnell. "Authentication of Users on Mobile Telephones - A Survey of Attitudes and Practices". In: *Computers & Security* 24(7) (2005), pp. 519–527.

[17] J. G. Cleary and L. E. Trigg. "K *: An Instance-based Learner Using an Entropic Distance Measure". In: *Proceedings of the 12th International Conference on Machine Learning (ICMLA)*. Vol. 5. Morgan Kaufmann, 1995, pp. 108–114.

[18] B. Coppin. *Artificial Intelligence Illuminated*. Jones & Bartlett Learning, Sudbury, 2004, pp. 267, 278, 283.

[19] T. Cover and P. Hart. "Nearest neighbor pattern classification". In: *Information Theory, IEEE Transactions on* 13(1) (1967), pp. 21–27.

[20] L. F. Cranor and S. Garfinkel. "Guest Editors' Introduction: Secure or Usable?" In: *Security & Privacy, IEEE* 2(7) (2004), pp. 16–18.

[21] I. Damousis, D. Tzovaras, and E. Bekiaris. "Unobtrusive Multimodal Biometric Authentication: The HUMABIO Project Concept". In: *EURASIP Journal on Advances in Signal Processing* 2008(1) (Jan. 2008), 110:1 –110:11.

[22] A. De Luca, A. Hang, F. Brudy, C. Lindner, and H. Hussmann. "Touch Me Once and I Know It's You!: Implicit Authentication Based on Touch Screen Patterns". In: *Proceedings of the ACM SIGCHI Annual Conference on Human Factors in Computing Systems (CHI)*. Austin, USA: ACM, 2012, pp. 987–996.

[23] M. O. Derawi, C. Nickel, P. Bours, and C. Busch. "Unobtrusive User-Authentication on Mobile Phones Using Biometric Gait Recognition". In: *Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*. Darmstadt, Germany, 2010, pp. 306–311.

[24] M. O. Derawi and P. Bours. "Gait and activity recognition using commercial phones". In: *Computers & Security* 39(B) (2013), pp. 137–144.

[25] R. Dhamija and L. Dusseault. "The Seven Flaws of Identity Management: Usability and Security Challenges". In: *Security & Privacy, IEEE* 6(2) (2008), pp. 24–29.

[26] J. Dorrier. "Sensors in Smartphones: Galaxy S4 Adds Pressure, Temperature, and Humidity Sensors". In: *SingularityHUB* (2013).

[27] A. P. Engelbrecht. *Computational Intelligence: An Introduction*. John Wiley and Sons, Chichester, 2007. Chap. 16.

[28] T. Fawcett. "An introduction to ROC analysis". In: *Pattern Recognition Letters* 27(8) (June 2006), pp. 861–874.

[29] T. Feng, Z. Liu, K.-A. Kwon, W. Shi, B Carbunar, Y. Jiang, and N Nguyen. "Continuous mobile authentication using touchscreen gestures". In: *Homeland Security (HST), 2012 IEEE Conference on Technologies for*. 2012, pp. 451–456.

[30] D. Florencio and C. Herley. "A Large-Scale Study of Web Password Habits". In: *Proceedings of the 16th International Conference on World Wide Web*. Banff, Alberta, Cananda: ACM, 2007, pp. 657–666.

[31] E. Frank. *J48*. URL: http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html.

[32] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song. "Touchalytics: On the Applicability of Touchscreen Input as a Behavioral Biometric for Continuous Authentication". In: *Information Forensics and Security, IEEE Transactions on* 8(1) (2013), pp. 136–148.

[33] S. Furnell. "An assessment of website password practices". In: *Computers & Security* 26(78) (2007), pp. 445–451.

[34] S. Furnell, N. Clarke, and S. Karatzouni. "Beyond the PIN: Enhancing User Authentication for Mobile Devices". In: *Computer Fraud & Security* 2008(8) (2008), pp. 12–17.

[35] S. Galatolo, M. Hoyrup, and C. Rojas. "Effective symbolic dynamics, random points, statistical behavior, complexity and entropy". In: *Information and Computation* 208(1) (2010), pp. 23–41.

[36] D. Goodin. "8 Million Leaked Passwords Connected to LinkedIn, Dating Website". In: *Ars Technica* (June 2012).

[37] Google. *2-Step Verification*. 2013. URL: http://www.google.com/landing/2step/?utm_campaign=en\&utm_source=en-ha-na-us-sk\&utm_medium=ha.

[38] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. "The WEKA Data Mining Software: An Update". In: *SIGKDD Explorations* 11(1) (2009), pp. 10–18.

[39] T. Himmelsbach. "A Survey on Today's Smartphone Usage". Diploma thesis. Technische Universität Berlin, 2011.

[40] C. C. Ho, C. Eswaran, K.-W. Ng, and J.-Y. Leow. "An Unobtrusive Android Person Verification Using Accelerometer Based Gait". In: *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia (MoMM '12)*. Bali, Indonesia: ACM, Dec. 2012, pp. 271–274.

[41] L. Hong. "Can Multibiometrics Improve Performance ?" In: *AutoID*. New Jersey, USA, 1999, pp. 59–64.

[42] A. K. Jain, A Ross, and S Prabhakar. "An Introduction to Biometric Recognition". In: *Circuits and Systems for Video Technology, IEEE Transactions on* 14(1) (2004), pp. 4–20.

[43] A. K. Jain, S. Prabhakar, and S. Pankanti. "On the similarity of identical twin fingerprints". In: *Pattern Recognition* 35(11) (2002), pp. 2653–2663.

[44] A. Kaltchenko. "Algorithms for Estimating Information Distance with Application to Bioinformatics and Linguistics". In: *Proceedings of Canadian Conference on Electrical and Computer Engineering (CCECE/CCGEI)*. Vol. 4. Niagara Falls: IEEE, May 2004.

[45] R. Kohavi. "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Proceedings of the 14th Iternational Joint Conference on Artificial Intelligence (IJCAI)*. Vol. 14. 2. Montreal, Quebec, Canada: Morgan Kaufmann, 1995, pp. 1137–1145.

[46] T. Kuseler, I. A. Lami, and H. Al-Assam. "Location-assured, multifactor authentication on smartphones via LTE communication". In: *Mobile Multimedia/Image Processing, Security, and Applications (SPIE)*. Vol. 8755. Baltimore, Maryland, USA, 2013, 87550B.

[47] LastPass. *The Last Password You'll Have to Remember*. 2012. URL: http://lastpass.com/ (visited on 06/14/2012).

[48] S. Lee and S. Zhai. "The performance of touch screen soft buttons". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2009, pp. 309–318.

[49] A. J. Lellimo. "PDAs - The Next Generation". In: *Network World* 12(11) (Mar. 1995), p. 66.

[50] L. Li, X. Zhao, and G. Xue. "Unobservable Re-authentication for Smartphones". In: *Network & Disctributed System Security Symposium (NDSS)*. San Diego, USA, 2013.

[51] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer New York, 2009.

[52] J. Lumsden. *Human-Computer Interaction and Innovation in Handheld, Mobile and Wearable Technologies*. IGI Global, 2011.

[53] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino. "Impact of Artificial "Gummy" Fingers on Fingerprint Systems". In: *Proceedings of SPIE* 4677(1) (2002), pp. 275–289.

[54] B Miller. "Vital signs of identity". In: *Spectrum, IEEE* 31(2) (1994), pp. 22–30.

[55] Y. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. "Unique in the Crowd: The privacy bounds of human mobility". In: *Scientific Reports* 3 (Mar. 2013).

[56] T. Mustafić, A. Messerman, S. A. Camtepe, A.-D. Schmidt, and S. Albayrak. "Behavioral biometrics for persistent single sign-on". In: *Proceedings of the 7th ACM workshop on Digital identity management (DIM)*. Chicago, IL, USA: ACM, 2011, pp. 73–82.

[57] NTT. *DoCoMo's Newest 505i Handset Features Fingerprint Authentication*. 2003. URL: https://www.nttdocomo.co.jp/english/info/media_center/pr/2003/000985.html (visited on 06/14/2012).

[58] V. Nannen. "The Paradox of Overfitting". Master's thesis. Rijksuniversiteit Groningen, 2003, p. 9.

[59] J. Nielsen. *Designing Web Usability*. Illustrate. New Riders, 2000.

[60] T. Oh, B Stackpole, E Cummins, C Gonzalez, R Ramachandran, and S. Lim. "Best Security Practices for Android, Blackberry, and iOS". In: *Enabling Technologies for Smartphone and Internet of Things (ETSIoT), 2012 First IEEE Workshop on*. 2012, pp. 42–47.

[61] Y. Park and J. V. Chen. "Acceptance and adoption of the innovative use of smartphone". In: *Industrial Management & Data Systems* 107(9) (2007), pp. 1349–1365.

[62] A. Peacock, X. Ke, and M. Wilkerson. "Typing Patterns: A Key to User Identification". In: *Security & Privacy, IEEE* 2(5) (2004), pp. 40–47.

[63] C. P. Pfleeger and S. L. Pfleeger. *Security in Computing*. 4th ed. Prentice Hall, Upper Saddle River, NJ, 2007, pp. 10–12.

[64] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA, USA, 1993.

[65] J. R. Quinlan. "Improved Use of Continuous Attributes in C4.5". In: *Journal of Artificial Intelligence Research* 4(1) (1996), pp. 77–90.

[66] J. R. Quinlan. "Induction of Decision Trees". In: *Machine Learning* 1(1) (1986), pp. 81–106.

[67] N. Sae-Bae, K. Ahmed, K. Isbister, and N. Memon. "Biometric-rich Gestures: A Novel Approach to Authentication on Multi-touch Devices". In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2012, pp. 977–986.

[68] H. Saevanee and P. Bhatarakosol. "Authenticating user using keystroke dynamics and finger pressure". In: *2008 International Conference on Computer and Electrical Engineering (ICCEE)*. Phuket, Thailand: IEEE, 2008, pp. 82–86.

[69] M. A. Sasse, S Brostoff, and D Weirich. "Transforming the Weakest Link - A Human/Computer Interaction Approach to Usable and Effective Security". In: *BT Technology Journal* 19(3) (2001), pp. 122–131.

[70] B. Schneier. "Two-Factor Authentication: Too Little, Too Late". In: *Communications of the ACM* 48(4) (2005), p. 27.

[71] B. Schneier. *Write Down Your Password*. June 2005. URL: https://www.schneier.com/blog/archives/2005/06/write_down_your.html (visited on 10/24/2013).

[72] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, S. Dolev, and C. Glezer. "Google Android: A Comprehensive Security Assessment". In: *Security & Privacy, IEEE* 8(2) (2010), pp. 35–44.

[73] K. Shah. *Top 10 iPhone Security Tips*. Tech. rep. McAfee, 2011, p. 3.

[74] M. Shahzad, A. X. Liu, and A. Samuel. "Secure Unlocking of Mobile Touch Screen Devices by Simple Gestures: You Can See It but You Can Not Do It". In: *Proceedings of the 19th annual international conference on Mobile computing & networking (MobiCom)*. Miami, Florida, USA: ACM, 2013, pp. 39–50.

[75] C. E. Shannon. "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27 (1948), pp. 379–423, 623–656.

[76] H. Sharp, Y. Rogers, and J. Preece. *Interaction Design: Beyond Human-Computer Interaction*. Wiley, 2007.

[77] D. Silverman. *Android 4.0s facial recognition is cool, but dont trust it yet*. 2012. URL: http://blog.chron.com/techblog/2011/12/android-4-0s-facial-recognition-is-cool-but-dont-trust-it-yet/ (visited on 06/14/2012).

[78] S. Sinofski. *Signing in with a Picture Password*. 2011. URL: http://blogs.msdn.com/b/b8/archive/2011/12/16/signing-in-with-a-picture-password.aspx (visited on 10/04/2012).

[79] D. K. Smetters and R. E. Grinter. "Moving from the Design of Usable Security Technologies to the Design of Useful Secure Applications". In: *Proceedings of the 2002 Workshop on New Security Paradigms (NSPW)*. Banff, Alberta, Canada: ACM, 2002, pp. 82–89.

[80] U. Topkara, W. Lafayette, and M. J. Atallah. "Passwords Decay, Words Endure: Secure and Re-usable Multiple Password Mnemonics". In: *Proceedings of the 2007 ACM Symposium on Applied Computing*. Seoul, Republic of Korea: ACM, 2007, pp. 292–299.

[81] A. Vance. "Simple Passwords Remain Popular, Despite Risk of Hacking". In: *NY Times* (Jan. 2010), A1.

[82] E. Vildjiounaite, S.-M. Mäkelä, M. Lindholm, V. Kyllönen, and H. Ailisto. "Increasing Security of Mobile Devices by Decreasing User Effort in Verification". In: *Second International Conference on Systems and Networks Communications (ICSNC)*. Cap Esterel, French Riviera, France, 2007, pp. 80–86.

[83] R. Wang, S. Chen, and X. Wang. "Signing Me onto Your Accounts through Facebook and Google: A Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services". In: *IEEE Symposium on Security and Privacy*. San Francisco, CA, USA: IEEE, May 2012, pp. 365–379.

[84] M. Weir, S. Aggarwal, M. Collins, and H. Stern. "Testing metrics for password creation policies by attacking large sets of revealed passwords". In: *Proceedings of the 17th ACM conference on Computer and communications security (CCS '10)*. Chicago, IL, USA: ACM Press, Oct. 2010, p. 162.

[85] A. Whitten and J. Tygar. "Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0". In: *8th USENIX Computer Security Composium*. Washington, D.C., USA: USENIX Association, 1999, pp. 169–184.

[86] J. D. Woodward, N. M. Orlans, and P. T. Higgins. *Biometrics*. 1st ed. McGraw-Hill/Osborne, Berkeley, 2003, pp. 186–187.

[87] J. Yan, A. Blackwell, R. Anderson, and A. Grant. "Password Memorability and Security: Empirical Results". In: *Security & Privacy, IEEE* 2(5) (2004), pp. 25–31.

[88] G. Yang, D. S. Wong, H. Wang, and X. Deng. "Two-factor Mutual Authentication Based on Smart Cards and Passwords". In: *Journal of Computer and System Sciences* 74(7) (2008), pp. 1160–1172.

[89] K. Yee. "Aligning Security and Usability". In: *Security & Privacy, IEEE* 2(5) (2004), pp. 48–55.

[90] N. Zheng, K. Bai, H. Huang, and H. Wang. *You Are How You Touch: User Verification on Smartphones via Tapping Behaviors*. Tech. rep. College of William & Mary Department of Computer Science, 2012.

[91] P. Zheng and L. Ni. *Smart Phone and Next Generation Mobile Computing*. Morgan Kaufmann, San Francisco, CA, USA, 2006.

Appendix A

Ethical Clearance

Clearance to perform the experiment discussed in this dissertation was obtained from the ethics committee of the Engineering, Built Environment and Information Technology Faculty at the University of Pretoria on the 4th of April, 2013. This appendix contains a copy of the permission letter granted by the committee.

Note that at the time of applying for ethical clearance, the project title differed from the title as it appears in the final version of the dissertation, but the protocol as approved was used unmodified.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Reference number: EBIT/02/2013 4 April 2013

Ms CJ Kroeze
130 Wicklow Avenue
Bronberrick
Centurion
0157

Dear Ms Kroeze,

FACULTY COMMITTEE FOR RESEARCH ETHICS AND INTEGRITY

Your recent application to the EBIT Ethics Committee refers.

1 I hereby wish to inform you that the research project titled "Using artificial immune systems for touch based intrusion detection on smart devices" has been approved by the Committee.

This approval does not imply that the researcher, student or lecturer is relieved of any accountability in terms of the Codes of Research Ethics of the University of Pretoria, if action is taken beyond the approved proposal.

2 According to the regulations, any relevant problem arising from the study or research methodology as well as any amendments or changes, must be brought to the attention of any member of the Faculty Committee who will deal with the matter.

3 The Committee must be notified on completion of the project.

The Committee wishes you every success with the research project.

Prof. J.J. Hanekom
Chair: Faculty Committee for Research Ethics and Integrity
FACULTY OF ENGINEERING, BUILT ENVIRONMENT AND INFORMATION TECHNOLOGY