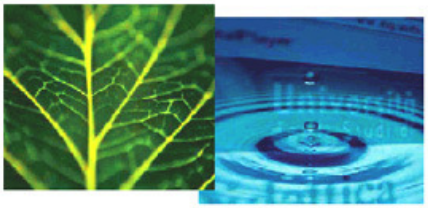
PHD DISSERTATION



International Doctoral School in Information Engineering and Communication Technologies (ICT), University of Trento, Italy.

Behavioral Biometrics for Smartphone User Authentication

Attaullah Buriro

SUBMITTED TO THE DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE (DISI) IN THE PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Advisor
Prof. Bruno Crispo, Università degli Studi di Trento, Trento, Italy.

Examiners
Prof. Fareed Melgani, University of Trento, Italy.
Prof. Javier Ortega-Garcia, Universidad Autonoma de Madrid, Madrid, Spain.
Prof. Nasir Memon, New York University Polytechnic School of Engineering, New York, USA.

Trento, Italy **28th February 2017**

Abstract

Pervasive in nature and extensively used for a wide range of features, smartphone provides functionality such as social networking, online shopping, mobile gaming, private/group communication, etc. While using these services, a user has to provide private information such as account credentials, credit card details, etc., which are then stored on the device. This information, if lost, can result in a user's privacy leakage and monetary loss. Therefore, significance of securing a smartphone from adversarial access becomes paramount. Despite being security and privacy critical, smartphones are still protected by traditional authentication mechanisms such as PINs and passwords, whose limitations and drawbacks are well known and well documented in the security community. The recent introduction of physical biometrics like facial, fingerprint and iris recognition, in smartphone authentication has mitigated the problems with user input, however, they still suffer from other usability and security issues. Hence, new, accurate, and user-friendly authentication mechanisms are required. In this direction, behavior-based authentication solutions have recently attracted a significant amount of interest in both commercial and academic contexts.

Most of the smartphone users prefer convenience over security and consider authentication mechanism more annoying as compared to other technological problems, such as lack of coverage, power consumption, etc. In this dissertation, we discuss limitations of existing authentication methods in terms of security and usability, and propose their replacements with behavioral biometric based authentication mechanisms. The underlying principle of our approach is to design solutions that authenticate users with either minimal or no cooperation from the users. We design, prototype and test the proposed authentication mechanisms based on our identified human behavior, such as how a person holds the phone, lifts the phone, types free-text PIN on the phone, signs her name on the touchscreen, etc. Moreover, we provide a comparative evaluation, based on accuracy, performance and usability, of our proposed mechanisms with the available state-of-the-art solutions. All of our solutions exploit the existing hardware (avoiding additional hardware requirement), and hence can be implemented on most of the smartphones available in the market today.

Keywords
[Smartphone, Biometric Authentication, Human-Computer Interaction, Sensors]

Acknowledgments

Alhamdulillah, all praises to Almighty Allah for His countless blessings throughout my life, in particular, to accomplish this achievement. I would like to thank my beloved father - who was my role model and my first mentor for his active support, motivation, and guidance during my early education. My mother for her unconditional love, prayers, and support. My wife for her incredible understanding and sacrifices on several occasions to my study requirements, i.e., deadlines, etc. throughout the degree period. My brothers and sisters for taking care of my mother, my wife and kids, in my absence and for their prayers. My dearest brother-like friend Maqsood Ahmad for his sincere, and unconditional willingness to help me overcome my shortcomings on several occasions.

During the four years as a Ph.D. student, I was lucky enough to meet amazing people who supported me a lot in various ways at the University of Trento. First and foremost, I would like to thank my supervisor, Professor Bruno Crispo for his excellent guidance and mentor-ship - I love the way he introduced me to research and shaped me as a researcher. I would always be grateful for his availability - he was always available to provide his insightful on the research challenges, I faced. I will always be thankful to him for his nice and easy way of expressing complex aspects of my research work and the efforts he made on co-authoring our papers. Second, my seniors, especially, Abdul Qadir Ahsan, Usman Raza, Mohammad Imran, Mohammad Rizwan Asghar, Tahir Khan, Talha Faizul Rehman, Ashad Mustafa, Mohammad Lamine, and Musawar Chowdhary who helped and encouraged me in the starting days of my PhD. I would also like to thank my group-mates Mojtaba Eskandri, Waqar Ahmad, and Filippo Del Frari, for their continued support and cooperation. A special thanks goes to Cristina Guerrero for her time to proof-read my thesis and helping me in preparation of this document.

I would like to take this opportunity to acknowledge EIT-Digital (MobileShield and SecurePhone Project), European Training Network for Cyber Security (NeCS grant number 675320), and NATO Communication and Information Agency, for partially supporting my PhD.

I would also like to thank the whole Unitn ICT administrative staff especially Andrea Stenico and Francesca Belton for their support and help in various aspects during my PhD study.

Dedication

To Baba, for his blessings, guidance and continued inspirational presence with me throughout my life.

To Amma, for her support, love, prayers and understandings of my feelings.

To brothers & sisters, for their prayers and love.

To Wife, for her support, help and care throughout my married life.

To my sweet kids - Fateh Muhammad Attaullah, Abdul Bais Attaullah and Abdul Basit Attaullah, for making my life sweeter.

Abbreviations

- **True Acceptance Rate** **TAR**
- **False Rejection Rate** **FRR**
- **False Acceptance Rate** **FAR**
- **True Rejection Rate** **TRR**
- **Equal Error Rate** **EER**
- **Weighted Error Rate** **WER**
- **Half Total Error Rate** **HTER**
- **Failure to Enroll Rate** **FTER**
- **Failure to Acquire Rate** **FTAR**
- **Receiver Operating Characteristics** **ROC**
- **Detection Error Trade-off** **DET**
- **Low Pass Filter** **LPF**
- **High Pass Filter** **HPF**
- **Dynamic Time Warping** **DTW**

Definitions

- **One-shot and Continuous User Authentication**: One-shot authentication schemes are designed to authenticate the user at the start of the session. In contrast, continuous authentication schemes are designed to continuously verify the user's identity throughout the entire session.
- **Implicit or Unobtrusive authentication**: The authentication schemes which don't require user's attention or cooperation for authentication purposes.
- **Intra-class Variations and Inter-class Similarities**: Intra-class variations refers to the observed differences in different samples acquired from the same user. In contrast, Inter-class similarities refer to the similarities found in the samples of different users.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

New generation devices, namely, smartphones and tablets, are the most widely used personal devices in everyday life. Currently, 400,000 Apple and 1.3 million Android devices are activated [6][7], while around 300,000 babies are born, each day. The usage patterns of smartphones are also very different than usage patterns of laptops and PCs. Smartphone users check their smartphone, on average, 150 times a day (once in every 6.5 minutes in 24 hours) [7]. Some of the reasons behind smartphone's popularity include their powerful processors, better batteries, improved hardware with powerful built-in sensors and faster connectivity chips. Apart from their hardware progression, developer-friendly operating systems have been continuously evolving and getting better since their first introduction by Apple in 2007 [8] and by Google in 2008 [9].

Widespread use of smartphones for broad range of activities poses serious security and privacy threats. In order to better obtain a clear picture of the threats to the user's data, a US-based security firm, Symantec, carried out a social experiment in five major cities in North America. They left 50 smartphones in public places without any protection [10]. Results revealed that 96% of who found the smartphones actually accessed them, and 86% of them accessed the personal information, 83% read business information, 60% opened social networking and personal emails, 50% started remote administration and 43% accessed online banking accounts [10].

Every smartphone available in the market today, continuously collects user's location coordinates keeps sending/receiving messages offers users capabilities for accessing mobile banking and social networking through the most popular apps such as Facebook, Whatsapp, Instagram, Viber, Twitter, etc. All of the mentioned apps store user's privacy sensitive data, which often becomes easily accessible once access to the phone is gained. Hence, any unauthorized access to these devices could have serious consequences and may become a nightmare for the victim [11].

1.1 Motivation and Problem Statement

The purpose of any authentication mechanism is to prevent any unauthorized access to the devices. The most widely used authentication schemes for smartphones are based on "*something the user knows*" (e.g., PIN/password), "*something the user possesses*" (e.g., some token), "*something the user is*" (e.g., face, fingerprint, etc.), and "*something the user does*" (voice, walks, etc.).

Authentication solutions based on what the user knows (PIN, password) are not considered to be highly secure anymore, and their associated security issues are well documented in the recent literature [12]. They are neither highly secure (since they are susceptible to guessing, shoulder surfing, and smudge attacks), nor highly usable (because they are frequently forgotten [13]. Further, keys, cards, and badges can be lost, or duplicated. Additionally, multi-factor (e.g., card + PIN, PIN + badges) also pose a usability issue: *why would a smartphone user carry an extra device for the sole purpose of authentication?*. As a result, a recent study reports that 70% of the smartphone users do not use any PIN/password [14], and they consider them more annoying than other technologically related problems, such as lack of coverage, small screen size or low voice quality [15].

To overcome the issues of PIN/password based authentication solutions, the focus of research has been diverted to biometric-based solutions. Apparently, this approach is well accepted in both academia and industry. For example, recent updates in smartphone authentication include face-unlock on Android platforms [16], voice recognition on Google smart-lock [17], and fingerprint unlock on iOS [18]. Recently, Google has announced to replace the passwords with their trust score based Trust API[1]. The Trust API will continuously monitor and keep calculating the trust in the user based on her available biometric data - keystrokes, location, etc. The idea is to increase the user's data security and privacy in a better, automatic, trustworthy and unobtrusive way.

Biometric authentication introduced by Bertillon in 1870s [19] is the process of verifying the identity of a person based on her biometric modalities or traits. Biometric modalities are broadly categorized as physical, behavioral, chemical and cognitive. Physical characteristics are based on the body parts, e.g., face, fingerprint, palm, iris, etc. Behavioral characteristics are based on behaviors, e.g., keystroke, gait, voice, etc., whereas the chemical characteristics are based on the events happening in the body, e.g., odor or temperature. Cognitive characteristics are based on the brain responses to specific stimuli, e.g., odor, sound, etc. Biometric authentication has multiple advantages over traditional authentication methods. Generally, they are considered more secure because they are hard to copy, and more reliable because they are hard to share or distribute and require the user presence at the time of authentication.

[1] http://www.networkworld.com/article/3074664/security/google-s-trust-api-bye-bye-passwords-hello-biometrics.html

The systems based on physical biometrics, such as face, iris and fingerprint have shown to be less preferred because of several reasons. Firstly, they require comparatively more user's cooperation since such biometric traits cannot be collected unobtrusively. Recent studies suggest that the user prefers convenience over the security, and that usability plays a major driver of user's adoption decisions [20]. More specifically, 47% of fingerprint and 36% of face recognition former users mentioned usability as the main factor to stop the usage of these technologies [21]. Secondly, the data acquisition time is significantly higher, i.e., iris$(15-20\text{sec})$[2], face$(6-10$ sec$)$. Face recognition struggles to perform in different lighting conditions, with the use of sunglasses, or other objects partially covering the face. Similarly, the quality of a voice sample is affected by different physical activities (e.g., when walking, climbing a mountain, running to catch the train red, etc.), also due to physical conditions (e.g., sickness). Thirdly, such systems can be spoofed[3],[4] [22, 23] and the incorporation of anti-spoofing technology [24] may increase the cost of the device.

It is well known that to perform a certain task every human employs different ways, methods, and knowledge. Behavioral biometrics work on the principle of "*how the user does something*", i.e., gait, keystroke, etc. Researchers have been working with different user behaviors such as their walking patterns (gait), the way of providing input (keystroke dynamics) [25, 26, 27] and the measurement of the arm movement [28, 29]. Behavioral biometrics offer many advantages over physiological traits. One of the main advantages is that the behavioral patterns can be collected transparently or sometimes even without user's knowledge. More importantly, data collection does not require any special dedicated hardware. However, most of the behaviors are not unique enough to provide accurate user identification but have shown promising results in user verification [30]. Since, behavioral biometrics are dependent on the user actions and habits, it makes them more attractive towards implicit user authentication.

1.2 Research Challenges

Behavioral-biometric-based authentication solutions have shown to be very promising. The main reasons behind their popularity include (i) the unobtrusive data collection, (ii) no need of additional hardware, (iii) some robustness against different environmental conditions, (iv) apparently very secure (because spoofing very private human behavior requires a lot of practice and time), and most importantly, (v) they may offer the revoking of the compromised behavioral attribute, unlike physical biometrics. However, since such solutions are quite new and less explored as compared to the physical biometrics, there

[2] http://www.ibtimes.co.uk/unlocking-phone-your-eyes-fujitsu-iris-recognition-tech-coming\-smartphones-2015-1490297

[3] https://www.theguardian.com/technology/2013/sep/22/apple-iphone-fingerprint-scanner-hacked

[4] http://www.iphonehacks.com/2016/02/iphone-touch-id-hacked-with-play-doh.html

exist a wide range of open challenges. The one addressed in this dissertation are discussed below:

1.2.1 Identification of Suitable Biometric Trait(s)

The selection of biometric modality(ies) mainly depends on the environment[5] and nature of the context in which the biometric system is implemented. The environment includes the feasibility and/or user acceptance of the characteristic in the target biometric system. Thus, the success of any biometric system largely depends on the selection of the right modality for the right application, i.e., user authentication on mobile devices.

The objective of this thesis is to investigate new behavioral biometric modalities that can be accurate in discriminating the users, robust against the possible attacks, and that can be collected transparently - without requiring extensive user cooperation for authentication. The motivation behind the search for new user behavior(s) is (i) to avoid any unnecessarily required user attention for the sole purpose of user authentication, which is very important for the usability of the mechanism, and (ii) to avoid any unnecessarily required hardware.

1.2.2 Uniqueness of Behavioral Patterns

Uniqueness/distinctiveness is considered as the key property of any biometric-based authentication system [2]: a measured trait/characteristic should be different enough from person to person. Higher uniqueness or distinctiveness results in higher accuracy. Physical traits are considered more distinctive both for verification and identification, whereas behavioral characteristics are considered sufficiently distinctive for verification purposes [30] only.

Thus, our focus in this thesis is to identify, prototype, and validate the unique human behaviors for user authentication on mobile devices. In the following chapters, we present how much our identified behaviors are sufficiently different for different users.

1.2.3 Limitation of Resources

Mobile devices come up with inherent computational and processing limitations, i.e., usually they are not rich in resources like desktops/laptops. Hence, the proposed mechanism(s) should have low processing complexity. Thus, any proposed authentication solution should be light-weight, rather than resource hungry, in order to attain wider usability and acceptability. They should be computationally inexpensive both in training and decision-making processes. Authentication solutions pose, both, one-time (in training) and run-time (testing) overheads on the normal operation of mobile devices.

[5] http://blog.m2sys.com/multimodal-biometrics-2/secret-on-choosing-a-suitable-biometric-modality/

Additionally, due to limited resources, applying blindly the features and machine learning algorithms may not be feasible/suitable for user authentication. A careful analysis of both features and algorithms is needed before their deployment on the real phone. The identification, extraction, and selection of an appropriate and productive set of features for behavioral biometrics (especially for smartphone user authentication) is still an open challenge. Similarly, selection of appropriate algorithms also needs careful evaluation, for example, applying advanced machine learning algorithms like Deep Neural Network (DNN) may raise performance related issues.

1.2.4 Data Collection

Behavioral biometrics, especially for smartphone user authentication, is a comparatively less-explored area. Most extensively tested/explored biometric modalities for user authentication are the keystroke or touch dynamics. Thus, we could only have keystroke or touch datasets available for evaluation. As we propose novel biometric authentication mechanisms, we are compelled to prepare our own datasets for the idea evaluation and prototyping.

1.2.5 Applicability to Different User Situations

Smartphone owners may use their devices in different positions or situations such as sitting, standing, walking, lying on the sofa and bed, walking upstairs and downstairs, jogging, driving, and cycling, etc. Therefore, while providing input sample, they need to hold the smartphone in such a way that the maximum screen becomes visible to them. Ideally, any proposed mobile biometric authentication mechanism should be situation and positions and activity independent (e.g., fingerprint recognition). Unfortunately, the majority of the proposed behavioral-based authentication solutions are limited to some specific activities and positions [27, 5, 31], hence, designing authentication solutions for all or most of the situations is still a challenging issue.

1.3 Our Approach

Our behavioral-biometric-based approach for smartphone user authentication starts with the collection of sensory data for the specific user movement and profiling the user based on the relevant information extracted from the collected sensory data. Our approach authenticates the legitimate user based on the similarity between the query sample and the stored samples.

Smartphones are full of different sensors (data sources), which can be used to record user behavioral patterns. Modern mobile operating systems provide developers with interfaces to collect sensor data and process it in their applications. Most importantly,

starting/stopping of such sensors does not require any user permission, as such, the user might not know about the data collection. We have registered various user movements with built-in 3-dimensional sensors available in most of the commercial smartphones in the market today. Registering and triggering smartphone sensors is easy and unobtrusive, i.e., prototype applications do not require any user permission(s).

Smartphone's 3-dimensional sensors generate continuous streams of data. These streams can be profiled as time series and later use for authentication purposes using time series analysis, however, considering the computational constraints, e.g., limited resources and computational time, this approach might not be practical and realistic. Alternately, a feature extraction scheme can be employed to extract the most useful/relevant features out from those recorded time series. We have applied feature extraction schemes in all of our proposed solutions.

Depending on the application context, a biometric system operates in two modes: identification and verification. In identification mode, the system recognizes an individual by comparing the captured biometric data with the templates of all the users in the database. Whereas, in verification mode, the user's identity is validated by comparing the captured biometric data with their own biometric template(s) in the database. In this thesis, we have focused on solving verification problem.

A biometric authentication/verification procedure is referred as a classification mechanism (see Figure 2.2). In particular, the matcher module is termed as the classifier. First, a specific classifier is trained on a dataset $\mathcal{D}$, consisting of samples over $(\mathrm{X},\ y)$; where X is variable with a set of attributes $\mathrm{X} = \{x_1 \dots x_n\}$ and y is the ground truth label. During operation, the trained classifier maps the input query to a certain class. In the case of binary classification, the training data contains the attributes and true labels of two classes and classifier, whereas, in 1-class verification, the training data comprises patterns of one user (owner) only. We have used binary-class classification for off-the-device analysis and $1 - class$ classification for on-board authentication on the smartphones.

1.4 Our Contributions

In this thesis, we present the design and implementation and technical details of new solutions for user authentication on smartphones. What makes our solutions unique and different than the existing ones is their minimal (or no) user effort requirement - our solutions authenticate their users with either minimal or without explicit user cooperation. In addition, all of our solutions exploit the existing hardware (avoiding additional hardware requirement), and hence can be implemented on most of the smartphone available in the market today.

The research contributions of this thesis are listed below:

- We introduce the *"Hold"* behavior (the way the user holds her phone in her hand), and show that the use of this biometric modality improves significantly the authentication accuracy and security.
- We propose light-weight, user-friendly, and power-friendly behavioral-biometric based solutions for smartphone user authentication.
- We report, in detail, the power consumption, processing overhead, usability, and security analysis of one of our proposed and implemented proof-of-concept mechanism and are in process of testing other prototype solutions.

1.5 Roadmap

Despite their lack of security and usability, knowledge-based authentication schemes (e.g., PIN and signature and others) are still used for one-shot authentication in smartphones as well as for accessing social networking sites, banking applications, etc. In order to make them more secure and usable, we added an extra transparent layer to these mechanisms. We transformed them to bi-modal systems, i.e., how a user moves the phone in her hands while entering the 4-digit PIN or signing with her fingertip on the touchscreen. The proposed bi-modal mechanisms based on "*Hold*" behavior and the way the user enters her secret or writes her name on the touchscreen are presented in Chapter 3 and Chapter 4, respectively. Then we present our fully unobtrusive unimodal authentication schemes, based on just "*Hold*" behavior, for both one-shot (Chapter 5) and continuous authentication (Chapter 6) on smartphones. Lastly, we present a fully unobtrusive tri-modal authentication scheme designed mainly to confirm the identity of the call picker. Normally, to pick up and answer a phone call, 3 actions are performed. Firstly, swiping the accept button, then bringing the device to the ear and start speaking. Our method, ITSME (Chapter 7) exploit these actions and verify the identify of the call picker.

1.6 Structure of the Thesis

`Chapter 2` focuses on introducing the background knowledge necessary to understand the biometric-based smartphone user authentication solutions.
`Chapter 3` presents a bi-modal system, i.e., how a user holds her phone in her hand and types a 4-digit *free-text* PIN/password. The use of phone-hold modality and the choice of 4-*digit free-text* differentiates *Touchstroke* from the classical key or touch stroke dynamics. *Touchstroke* transparently collects phone-movements in the background and provides the flexibility of entering any 4-*digit free-text* making it widely acceptable among the smartphone users.

`Chapter` 4 is a variant of *Touchstroke* and exploits the user signing style (how the user writes her name on the touchscreen), in addition to the user's hold behavior. *Hold & Sign* does not take into account the signature image, rather the finger movements on the touchscreen and the in-hand phone movements. *Hold & Sign* provides two benefits: firstly, it enhances usability by providing its user the flexibility of writing user's own name. Secondly, it increases security - it becomes extremely difficult to mimic finger and hand movements at the same time.
`Chapter 5` introduces a completely unobtrusive uni-modal system based on user's natural hand movements. The mechanism starts profiling user's hand micro-movements after an unlock event is notified. Our solution is completely implicit and is applicable both for a smartphone with and without any enabled authentication mechanism. An attacker has to pass this authentication mechanism too besides the other authentication requirement, e.g., passcodes, etc.
`Chapter 6` presents *ACTIVEAUTH* - a fully unobtrusive motion-based one-shot-cum-continuous user authentication scheme for smartphones, which in addition to authenticating the user at login stage (Chapter 5), continuously tracks the user interactions and authenticate the user before a package is installed/uninstalled. Besides providing a one-shot login (as in Chapter 5), our approach determines who should be allowed to install a new application package or uninstall an already installed package. *ACTIVEAUTH* can be implemented as a standalone scheme or can be augmented with any of the existing scheme to strengthen its robustness against the possible attacks.
`Chapter 7` presents a tri-modal authentication solution based on sliding, lift behavior and voice modality. Sliding means the way the user drags the start button to slide-unlock the smartphone and lift behavior refers to the movement the user makes to bring the phone towards her ear. We implemented and tested this solution on a real Android phone and evaluated it in multiple user situations.
`Chapter` 8 concludes the thesis and presents the possible future work emerging from this work.

Chapter 2

Background

In this chapter, we present the background knowledge necessary to understand the smartphone user authentication problem.

2.1 Introduction

Human biological data, due to its permanence and uniqueness, can be used as a means of identification, authentication and access control. The use of biological data for the purpose of identity management is termed as biometric recognition or simply biometrics. Physical (based on the physical characteristics) and behavioral (based on behaviors) biometrics are the most popular types. Physical characteristics include fingerprints, hand geometry, iris or retina scans, etc., and behavioral characteristics include keystrokes, gait, signature, voice, etc., (see Figure 2.1). Other biometrics use chemical features (based on events that happen in a persons body, measured by e.g., odor or temperature) and cognitive features (based on brain responses to specific stimuli, e.g. sound).

Biometrics has got all the potential to completely replace knowledge-based solutions, because their alphanumeric counterparts can be stolen, forgotten, and shared. Biometric authentication has been studied for a long time. Large-scale commercial deployments already exist, such as the fingerprint sensors on laptops and smartphones. However, these deployments are based on physical biometrics, which essentially require explicit user action, hence annoying the users [32] and provide a "*one-shot*" authentication.

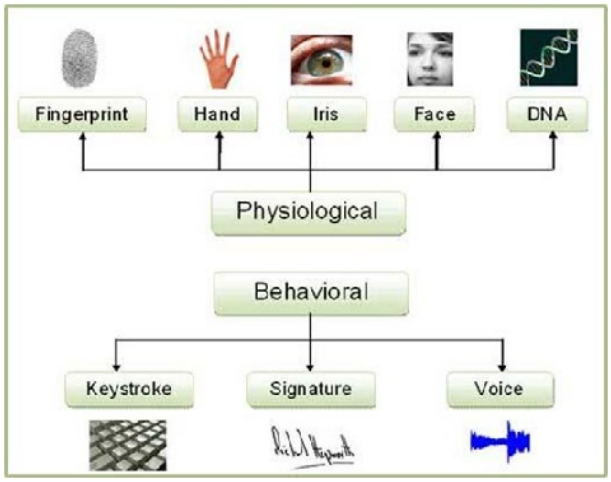
Most of the research about transparent, implicit and continuous authentication has been done in smartphone's security and access control is based on behavioral biometrics. Behavioral biometrics offer many advantages over physiological traits. One of the main advantages is that the behavioral patterns can be collected transparently or sometimes even without user's knowledge. More importantly, data collection does not require any special dedicated hardware. However most of the behaviors are not unique enough to provide accurate user identification but have shown promising results in user verification

[30]. Various behavior-based authentication solutions have been tested and evaluated [33, 29, 31] but are yet to be deployed at large scale. One reason is that the performance of many of these schemes is not yet at the same level as physical biometrics. Another reason is that, not much attention has been paid to the performance of biometrics under differing or difficult circumstances. For example, gait authentication is typically evaluated by having subjects walk along flat surfaces of corridors in buildings.

Recent literature categorizes human behaviors into 5 different kinds on the basis of data collection method [30], as follows:

- Authorship based Biometrics:
  - User verification/identification on the basis of the way a user writes or makes drawings on a piece of paper.
- Human Computer Interaction:
  - Different users employ different strategies, different styles, and differently apply their abilities and knowledge in everyday interaction with computers and new generation devices. These traits yield sufficient features for successful user verification/identification.
    - *HCI-through Input Devices*:The kind of HCI where human interaction is made through input devices, such as keyboard, mouse etc.
    - *HCI-based Behavioral Biometric*: HCI-based behavioral biometrics which measures advanced human behaviors such as strategy, knowledge or skills exhibited by the user during interaction, through different softwares.
- In-Direct HCI-based Biometric:
  - Measurements obtained by monitoring user's HCI behaviors indirectly via low level actions of computer software. Identification through audit logs [34] and registry access [35], etc.
- Motor skills (kinetics):
  - it is an ability of a human to utilize the muscles [35]. As these movements rely upon the proper functioning of brain, joints, skeleton and nervous system, they indirectly reflect the quality of functioning of such systems, making verification possible.
- Purely Behavioral Biometrics:
  - it measures human behaviors not directly concentrating on measurements of body parts or intrinsic, inimitable and lasting muscle actions, such as the way an individual walks or types [30].

Figure 2.1: Types of physical and behavioral biometric modalities [2].



2.2 Biometric Recognition System

Any biometric recognition system (see Figure 2.2) is bound to automatically identify a person by examining some already enrolled physical and/or behavioral characteristics with its corresponding query characteristics submitted by that person. An ideal biometrics is supposed to have "*zero*" false acceptance and false rejection rates, and should satisfy some properties (see Section 2.4), such as universality, uniqueness, permanence, acceptability, and should be robust against possible attacks.

Figure 2.2 depicts the block diagram of a biometric system with four components, defined below:

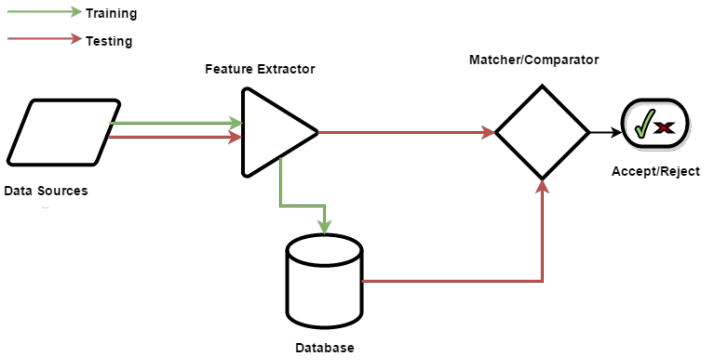
- **Data Source**: This block deals with the biometric data capture from an individual. It includes both the hardware and software. It may also incorporate an additional "*Quality Checker*" component, to ensure the data quality.

- **Feature Extractor**: This block deals with the extraction of discriminatory features from the captured biometric sample in order to profile the most relevant user information in the database.

- **Database**: This block deals with the storage and management of the biometric template generated from the user's data.

- **Matcher/Comparator** This block matches the claimed or query pattern with the earlier stored pattern(s), and decides in terms of acceptance/rejection. For verification, it performs the one-to-one match, and for identification it matches the input

Figure 2.2: A Generalized Biometric Recognition System.



query pattern with all the stored patterns of all the classes (1:N), after which the user is identified based on the highest achieved score.

2.3 Biometric Verification Vs. Identification

Authentication or verification refers to the identity confirmation. When a users claims an identity (e.g., by inserting card into the ATM machine, or inserting card to access secure facilities, and then typing a password or PIN), the system performs certain computations to confirm the claim of the user. This comparison of claim with already stored template is referred to as 1:1 match.
Identification differs from verification, where the unknown query template comes from a known user and the job of the system is to correlate it with a known user's template to which it corresponds. This process is termed as 1:N matching. Identification can further be classified into two kinds: open-set and closed-set. The identification is closed set if the template of the users being verified already exists in the classifiers database; otherwise, it is termed as open-set identification.
In this thesis, we have focused on solving authentication problem.

2.4 Choice of Biometric Traits

The choice of biometric modality(ies) depends upon various issues besides their recognition performance. The literature [2] considers seven factors for determining the suitability of these traits as discussed below:

- **Universality**: Every user needs to have that biometric modality. This factor helps in determining (Failure to Enroll Rate (FTER) of biometric recognition system.

- **Uniqueness**: The given modality should be sufficiently different across individuals in a set of population.
- **Consistency**: The given modality should be consistent over a certain time period.

- **Measurability**: The possibility to acquire and digitize the biometric modality with best devices without causing any inconvenience to the user.
- **Performance**: Besides recognition accuracy, the throughput of the biometric system should also cope with the constraints imposed by the application.
- **Acceptability**: It reflects the ease and comfort with which users provide their traits to the system.
- **Circumvention**: It refers the ease with which the modality of other participants is copied, imitated or modified to gain illegitimate access of the system.

2.5 Multi-Modal Biometrics

Recent years have witnessed a significant increase in accuracy and reliability in biometric authentication. However, mostly evaluated and tested advance biometric systems also have some limitations; some of these limitations are related with type of data, and some are related with methodology. More specifically, performance of the biometrics systems suffers a lot due to the presence of noise in input data, inter-class variations, non-universality, and other possible factors that may affect the performance, security and usability of those systems [36].

A multimodal biometric system is a newer way to address some of the problems associated with unimodal biometric systems. It incorporates the consolidation of data presented by multiple information sources. Multimodal systems can significantly improve recognition performance along with increase in population coverage (thus reducing FTER), prevents spoof attacks, and increase the degree of freedom. Although these systems require more storage, take higher processing time, and involve more computational cost as compared to unimodal biometric systems, the above mentioned advantages are compelling for their deployment in large scale authentication systems [37].

2.6 Smartphone Sensors

Mobile sensors are broadly categorized in three types, i.e., motion sensor, position sensors and environmental sensors [3]. Motion sensors (accelerometers, gyroscope, gravity sensor, etc.) measure the acceleration and rotational forces along three axes. Position sensors (orientation and magnetometer sensors) measure the physical position of the smartphone.

Environmental sensors, e,g., barometers, thermometers, etc., measure various environmental parameters. Motion and position sensors have shown to be accurate in discriminating the users and have widely been used for smartphone user authentication [27, 31, 5]. Environmental sensors may not be useful for "*one-shot*" authentication however, they could better be used for continuous authentication. We have used only position and motion sensors in our proposed solutions.

In Android mobile operating system, data can be collected in both fixed and customized intervals after registering the sensor with *registerlistener()* termed as *Sensor Delay Modes*[3].

Android supports four fixed intervals, namely, *SENSOR_DELAY_FASTEST* without any delay in throwing samples, *SENSOR_DELAY_GAME* with a fixed delay of 20,000 μ seconds, *SENSOR_DELAY_UI* with a fixed delay of 60,000 μ seconds, and *SENSOR_DELAY_NORMAL* with a fixed delay of 200,000 μ seconds.

We have tried all delays for the data collection, however, *SENSOR_DELAY_GAME* provided better accuracy so we use *SENSOR_DELAY_GAME* for acquiring sensor values from all the sensors. Additionally, *SENSOR_DELAY_FASTEST* is highly likely to include noise in the data. Similarly in *SENSOR_DELAY_NORMAL* and *SENSOR_DELAY_UI*, it is quite possible that some of the sensors might not sense the user interaction correctly (e.g., in some of our experiments orientation sensor in both delays could not sense the user interactions).

We explain below the working principle of our chosen sensors:

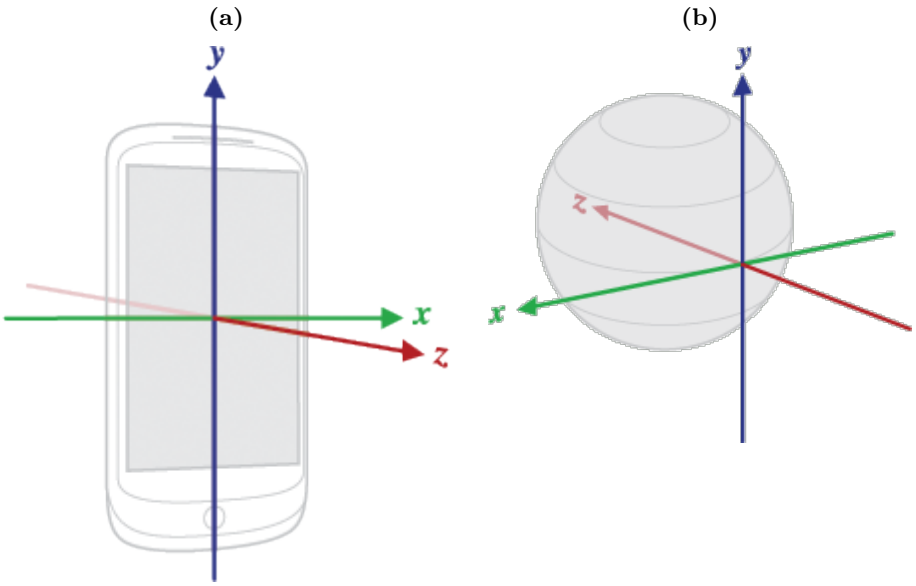
2.6.1 Accelerometer Sensor

This sensor measures the *acceleration* applied to the device, including the force of gravity, measured on three axis' X, Y and Z. Android's sensor API uses a standard three-axis coordinate system. This system is defined relative to the device's screen when it is held upright as shown in Figure 2.3a. The acceleration that is applied to a device A_d is calculated using the forces (including gravity g) that are applied to the sensor F_s itself using the following equation:

$$A_d = -g \sum \frac{F_s}{mass} \tag{2.1}$$

In order to remove the contribution of the force of gravity for the raw accelerometer data, we applied High Pass Filter (HPF) and obtained HPF accelerometer readings. The motivation was to obtain the exact acceleration applied by the user on the device. Conversely, we applied the Low Pass Filter (LPF) to raw accelerometer data to obtain the apparent transient forces acting on the device, caused by the user activity. These two sensory readings (HPF and LPF) can be obtained by applying a filter constant (α, we

Figure 2.3: (a) Coordinate system relative to the device [3]. (b) Coordinate system used in orientation sensor.



used 0.5). This filter constant is calculated from the estimation of the latency (the filter adds to obtain the sensor events) and the actual sensor event delivery rate[1]. Thus, we have used 3 variants of the accelerometer, i.e., Raw, HPF and LPF, in our analysis.

2.6.2 Gravity Sensor

This sensor measures the applied force of gravity (m/s^2) on the smartphone in three dimensions. In simple words, it provides magnitude and direction of the force of gravity applied on the phone. The coordinate system and the unit of measurement of gravity sensor are the same as of the accelerometer sensor.

2.6.3 Gyroscope Sensor

This sensor measures the smartphone rate of rotation (rad/s) in three dimensions.

The sensor's coordinate system is the same as the one used for the acceleration sensor. The counter-clock-wise rotation is positive, i.e., an observer if looking from some positive location on the three axes at a device positioned on the origin world, is considered positive.

[1] `http://developer.android.com/guide/topics/sensors/sensors_motion.html`

2.6.4 Magnetometer Sensor

The magnetometer sensor measures the strength and/or direction of the magnetic field (μT) in three dimensions. It differs from the compass as it does not provide point north. The magnetometer measures the Earth's magnetic field if the device is placed in an environment absolutely free of magnetic interference.

2.6.5 Orientation Sensor

This sensor computes the values of the different angles representing the orientation of the smartphone in three axis. It records the azimuth, pitch and roll in three dimensions. Specifically it shows the mode (portrait or landscape) of the phone. Note that the orientation uses a different coordinate system than the accelerometer and the gyroscope, as depicted in the Figure 2.3b.

- X is defined as the vector product $Y \cdot Z$ (it's tangential to the ground at the device's current location and roughly points West).
- Y is tangential to the ground at the device's current location and points towards the magnetic North Pole.
- Z points towards the center of the Earth and is perpendicular to the ground.

There is a strong relationship between the wrist motion and the readings of orientation sensor when a user holds and operates her phone. Human wrist provides 3 dimensions of freedom. In medical terms, these 3-dimensions are referred as wrist flexion and extension, the supination and the pronation, and the wrist radial and ulnar deviation [38, 39].

Wrist extension is the upward movement of the wrist causing palm facing outward and flexion is the opposite of it, i.e., it is the downward and inward movement of the wrist causing palm facing inward. This movement causes variation in pitch direction (y-axis) on orientation sensor. Supination is the way the arm rotates when the palm faces forward. Pronation is the way of rotating forearm when the palm faces backward. This movement causes a variation in the roll direction (z-axis) of the orientation sensor. Lastly, radial and ulnar deviation are the side-to-side movement of the hand at the wrist, toward or away from the thumb. The movement in this dimension corresponds to the azimuth direction (x-axis) of the orientation sensor.

2.6.6 Touchscreen

The touchscreen provides the user interface for the operation of the device. Devices can be categorized as single and multi-touch devices. Finger and/or a pen acts as a tool to interact with the touch screen. In Android, the library *MotionEvent* provides a class for

tracking the motion of different pointers such as fingers, stylus, mouse, trackball, etc. This event, triggered as a result of a touch, is reported by an object of this class. This object may contain a specific action code like the location of the touch on XY coordinates of the touch screen, information about pressure, size and orientation of the touched area. Action code represents the state of the touch action, e.g., `Action_Down` stands for the start of a touch action while `Action_Up` represents the end of a touch action. The Android `VelocityTracker` class is used to track the motion of the pointer on the touch screen. The class methods, `getXVelocity()` and `getYVelocity()`, are used to acquire the velocities of the pointer on the touch screen in X and Y axis respectively.

2.7 Classification

Generally, the problem of user authentication is solved in two ways: with binary classification (training with two classes) and anomaly detection (training with only one target class). Classifiers are very powerful in discriminating the true user from a given training set, whereas anomaly detectors actually check for deviation from the legitimate user's behavior and authenticate/reject on the basis of this deviation.

Binary class classification might be suitable/acceptable for off-the-device analysis (for an initial assessment and understanding), however, 1-class verification is considered as more practical and realistic approach for the implementation of such systems for smartphones [31, 40, 5]. The primary reason of dealing with authentication (on smartphones) as 1-class verification is, since the binary classifier requires biometric data from both the owner and non-owner, hence sharing such biometric information among the smartphone users may lead to privacy concerns.

2.8 Success Metrics

In this section, we explain our success metric. The results of our evaluations are presented using multiple terms explained below:

- **True Acceptance Rate (TAR)**: is the fraction of positive samples correctly classified as positives.
- **False Acceptance Rate(FAR)**: is the fraction of the negative samples incorrectly classified as positives.
- **False Rejection Rate (FRR)**: is the fraction of the positive samples incorrectly classified as negatives.
- **True Rejection Rate (TRR)**: is the fraction of the negative samples correctly classified as negatives.

- **Equal Error Rate (EER)**: is the value where FAR and FRR become equal.
- **Weighted Error Rate (WER)**: is the combined error rate of both FAR and FRR with a weight α assigned to each.
- **Half Total Error Rate (HTER)**: As proposed by Poh et al. in [41], the final evaluation looks at the performance of the system after deciding on the weight α and the optimal threshold Δ_α^*. This is measured by the so called Half Total Error Rate (HTER), which is calculated as follows:

$$FAR(\Delta) = \frac{FA(\Delta)}{nI} \tag{2.2}$$

$$FRR(\Delta) = \frac{FR(\Delta)}{nG} \tag{2.3}$$

Given a specific threshold Δ, the FAR is defined as the number of false acceptances (FA) divided by the number of imposters nI and the FRR is defined as the number of false rejections (FR) divided by the number of genuine users nG.

To evaluate the interaction of these error rates the Weighted Error Rate (WER) is used. The WER shows the combined error rate of both FAR and FRR with a weight α assigned to each. If the false accepts are considered worse than false rejects (focus on security), a weight > 0.5 should be used. If false rejects are worse than false accepts (focus on usability), than a weight < 0.5 is more appropriate. A special error rate is the EER where both errors have the same weight (i.e. $\alpha = 0.5$). The WER is defined [41] as follows:

$$WER(\alpha, \Delta) = \alpha FAR(\Delta) + (1 - \alpha)FRR(\Delta) \tag{2.4}$$

Given a specific weight α, the goal is to find the optimal threshold Δ_α^* for which the WER is as low as possible. This function can be defined as:

$$\Delta_\alpha^* = \underset{\Delta}{\operatorname{argmin}} \left|\alpha FAR(\Delta) + (1 - \alpha)FRR(\Delta)\right| \tag{2.5}$$

$$HTER(\Delta_\alpha^*) = \frac{FAR(\Delta_\alpha^*) + FRR(\Delta_\alpha^*)}{2} \tag{2.6}$$

The lower the HTER, the better the system performs given the chosen weight α.

- **Accuracy**: It is the ratio of correct assessments to all the assessments.

$$Accuracy = \frac{TPR + TRR}{TPR + FPR + FNR + TNR} \tag{2.7}$$

- **Failure to Acquire Rate (FTAR)**: The proportion of failed recognition attempts (due to system limitations). A reason for this failure could be the inability of the sensor to capture, insufficient sample size, number of features, etc.
- **Receiver Operating Characteristic (ROC) Curves**: ROC is plotted against the TAR and FAR. ROC curve starts from the (0,0) coordinates and pass through the (1,0) coordinates and finishes at the (1,1) coordinates. The curve closer to the (1,0) coordinates reflects a better performance.
- **Detection Error Tradeoff (DET) Curves**: DET is used to show the correlation between the two common error types, i.e., FAR on the x-axis and FRR on the y-axis. This curve starts from (0,1) coordinates, passes through (0,0) coordinates and finishes at (1,0) coordinates. The curve closer to (0,0) coordinates indicate a better performance.

2.9 Chapter Summary

In this chapter, we have presented the necessary background to understand the problem and the proposed solutions. We start with the general introduction of biometrics, the existing different types of them, and the reasons for choosing behavioral biometrics to solve the problem. We explain the classical biometric recognition system. We elaborate on the important factors behind the choice of a biometric modality. We have discussed in details how we register user movements and finger movements (touch-based features) using 3-dimensional built-in sensors and the smartphone touchscreen. Our approach leverages the most commonly available built-in smartphone unprivileged sensors (sensors are started/stopped without requiring any user permission(s)). We relied on `SENSOR_DELAY_GAME` in most of the proposals, because it proved itself as the most reliable and useful sample rate. We also describe the classification model and success metrics used to evaluate our proposed solutions.

Chapter 3

Touchstroke:Touch-typing Based Smartphone User Authentication

In this chapter, we propose a new bi-modal biometric authentication solution, *Touchstroke*, which leverages on user's hand movements while holding the device and the timings of touch-typing (the act of typing input on the touchscreen of a smartphone) when the user enters *text-independent* 4-digit PIN/password. *Touchstroke* exploits the most commonly available hardware, i.e., sensors, without the need of any additional hardware, making itself usable in any off-the-shelf commercially available smartphone.

Initial experiments with state-of-the-art classifiers prove *Touchstroke* handy in each user situation. Preliminary results are encouraging, showing higher accuracy, thus, making *Touchstroke* a plausible alternative to traditional authentication mechanisms.

The part of this chapter is published in [27]: Attaullah Buriro, Bruno Crispo, Filippo Del Frari, and Konrad Wrona. Touchstroke: smartphone user authentication based on touch-typing biometrics, in proceedings of the International Conference on Image Analysis and Processing pp. 27–34, Springer, 2015.

3.1 Introduction

This chapter presents a new behavior-based authentication scheme called *Touchstroke*, which leverages two human behaviors: how the phone is held and how a 4-*digit text-independent* PIN/password is entered. Our experiments confirmed that every user has a unique phone movement behavior and a different way of touch-typing a PIN/password on the smartphone. *Touchstroke* computes the phone holding behavior with built-in 3−dimensional smartphone sensors: orientation, gravity, magnetometer, gyroscope and 3 variants of the accelerometer. Sensors are started at the time of the first touch-type and stopped after the fourth and final touch-type. Users are allowed to input any combination of 4-*digit* numbers and/or alphabets, hence they are expected to be quite comfortable

while using this authentication mechanism.

We extracted 4 statistical features from each data stream from all the physical sensors (total 16 from each sensor) and 14 features related with *n-graph*, namely dwell time and flight time (see Figure 3.1), from each typing pattern. In a study [4], authors show that these features are the most widely used features in keystroke dynamics. In order to check the usability of our proposed method, we collected 30 observations from 12 users in 6 significantly different activities.

The purpose of this chapter was to check the efficacy of each individual built-in 3-dimensional sensor towards the user authentication and the analysis was performed off the smartphone so we dealt this as a binary-class classification problem.

We tested our dataset using two state-of-the art binary classifiers, BayesNET and Random Forest (RF). The reason behind this selection is that they have shorter computation time and resistance against over-fitting.

3.1.1 Contributions

The main contribution concerning this section are listed below:

- The proposal of *Touchstroke*- a novel behavioral biometric user authentication mechanism, based on how the user *holds* her smartphone and enters her 4-*digit* secret *free text* on the smartphone touchscreen.
- Experimental validation, proving every built-in sensor worthy to be used for smartphone user authentication. More specifically, the data generated from each source, while the user enters her secret free-text, illustrates the importance of each sensor.
- The collection and sharing of data from multiple sensors in multiple user situations from 12 users. Our collected dataset contains 30 patterns in each of the 6 activities (30 x 6 x 12 = 2,160 patterns from each sensor).

3.2 Related Work

Keystroke-based user authentication is the mostly evaluated and tested behavioral biometric method for user authentication on PCs and smartphones using hardware and software keyboards. Since, we have implemented text-independent touch-typing dynamics using Android soft-keyboard, we consider soft-keyboard-based work as our related work.

3.2.1 Software Keyboard-based User Authentication

Keystroke-based recognition systems employ measurement of the user's typing behavior on digital input devices such as smartphones and tablets. A digital signature is prepared

on the basis of a user's interactions with these devices. Specifically, a user is asked to provide an alpha-numeric PIN/password to the system for creating a template for training and later for testing. The studies [4, 42] suggest that this fingerprinting is fairly unique from person to person thus can be used as a base for user identification.

A study conducted by Huang et al. [43] explored soft keyboard-based user authentication on mobile phones. The users were asked to enter their names and passwords 6 times for training. Based on the keystroke latency and key-hold-time features, they achieved an Equal Error Rate (EER) of 7.5%.

Saevancee and Bhattarakosol [44] reported an EER of 1% using the K-Nearest Neighbour (KNN) algorithm and reported similar results using neural networks [45]. However, they conducted their experiments only using a notebook touchpad. A recent study conducted by Saira et al. [46], on smartphones, revealed that the keystroke pressure might not be unique and hence ended up with an EER of 8.4% when used in conjunction with classical keystroke features (timings).

3.2.2 Sensor-assisted Keystroke-based User Authentication

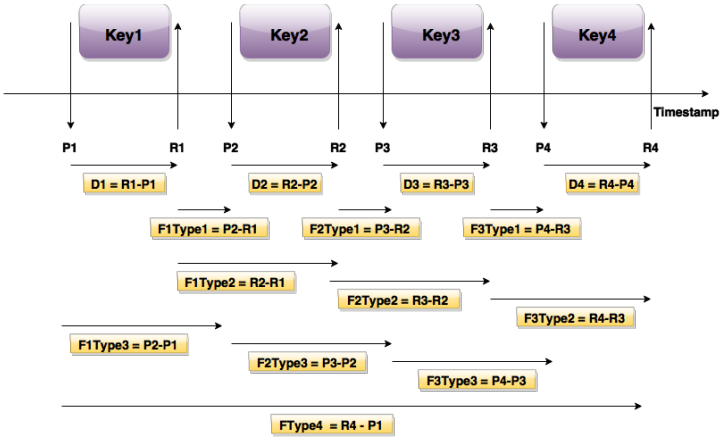
Recent literature reports the feasibility of using sensor data in combination with keystrokes for user authentication.

Several projects have been conducted to study the use of accelerometers and gyroscopes. For example, Giuffrida et al. [42] introduced *UNAGI*, a *fixed-text* and sensor-enhanced authentication mechanism for Android phones. They evaluated their method with 20 subjects and achieved an EER of 4.97% for passwords, and 0.08% for only sensor data. Miluzzo et al. [47] used sensor data to infer the icon activated by the user of iOS devices and reported 90% accuracy.

Similarly, Aviv et al. [48] presented a method that relies on accelerometer data and keystroke timings to infer 4-digit PINs for unlocking smartphones. Specifically, they demonstrated the use of accelerometer data for learning user tapping and gesture-based inputs as these methods are required to unlock smartphones using PIN/password and graphical password patterns. Additionally, they collected data in two situations, *sitting* and *walking*.

Touchstroke is different from the previous solutions in terms of features (for sensors), classification strategies, number of sensors, sensor-data-acquisition and constraints on the input.

Figure 3.1: Touchstroke features used in this paper [4].



3.3 Approach

In this section, we illustrate the main approach adopted by our solution.

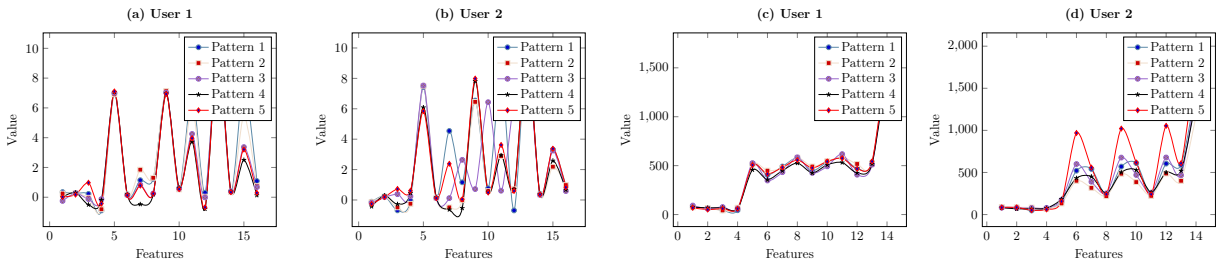
3.3.1 Intuition Assessment

It has now become a fact that each each user has a different way of holding and moving the phone when entering his PIN/password [27] [31]. An adversary can spoof and copy what is being written but it is very difficult to copy the exact timings of touch-types.

Our intuition is correct if and only if the patterns of the same user are very similar (high intra-class similarity) and patterns of different users are different enough (high inter-class variations).

We argue (see Figure 3.2) that the patterns of the same user are very similar to each other and patterns of two users are different enough. We show the patterns of raw accelerometer and touchstroke sensor data for a single situation: when the user is *sitting*.

Figure 3.2: Comparison of 5 patterns of Raw accelerometer (a and b) and touchstroke data (c and d), in *sitting* position for two users.



3.3.2 Our Solution

Our approach is based on the profiling of the user's hand micro-movements while the user enters her 4-*digit* secret *free-text*, i.e., PIN/passwords. We deal this as a bi-modal system, i.e., first modality is based on the differences in the keystroke timings (we call touch modality) and second is based on the *hold* behavior, i.e., the way user holds her phone in her hand(s).

3.3.3 Considered Sensors and Classifiers

Our solution makes use of five 3-dimensional sensors: the accelerometer (Raw, LPF and HPF); the gravity; the magnetic field or magnetometer; the gyroscope; and the orientation.

All the above sensors generate continuous streams in X, Y and Z directions. We have added a fourth dimension to all of these sensors and name it *magnitude*. Magnitude has been tested in the context of smartphone user authentication [27][49][31] and has proved to be very effective in classification accuracy. The magnitude is mathematically represented as:

$$S_M = \sqrt{(a_x^2 + a_y^2 + a_z^2)} \tag{3.1}$$

where S_M is the resultant dimension and a_x, a_y and a_z are the accelerations along the X, Y and Z directions.

Classifier selection depends on type and size of the dataset. We selected two classifiers by considering their short computation time and their resistance against over-fitting. Normally, BayesNET classifier works well on small datasets and a RF classifier is equally good for small and large datasets. We have used these classifiers (with default parameters) in portable GUI-based WEKA Experimenter Workbench.

3.4 Experimental Evaluations

In order to validate our initial intuition we ran a series of experiments, described in the sections below:

3.4.1 Data Collection

We implemented *Touchstroke* as an Android application that triggers all physical sensors from the first touch-type and stops them after the last touch-type. At this moment the app is designed for only four touch-types with the possibility to be extended. We recruited 12 volunteers for our experiment; most of them are either Master's or PhD students but

not security experts. In order to check the effectiveness of our proposed mechanism, we collected data in six different user positions, i.e. *sitting, standing, walking, lying on sofa, walking upstairs and walking downstairs*. We used a Google Nexus 5 running KitKat 4.4.2 for data collection. We collected 30 patterns from each user in each activity. In total, we collected 180 samples (in all 6 activities) from each user, making a total of 2160 samples per sensor from 12 users.

Touchstroke collects sensor data in *SENSOR_ DELAY_ GAME* mode.

3.4.2 Feature Extraction

We have four data streams from every 3-dimensional sensor. We chose statistical features because it is computationally cheaper to compute them. We extracted 4 statistical features, namely mean, standard deviation, skewness and kurtosis from each data stream. In this way, data from every sensor is transformed into a 4 by 4 feature matrix. Thus, we have 16 features from all four dimensions of each sensor. Similarly, we extracted 14 features (see Figure 3.1), based on touch-typing timing, from the *text-independent* 4-*digit* PIN/password entered by the user.

3.4.3 Data Fusion

Data fusion can be done at the sensor level, feature level, match score level, rank level and decision level. Data fusion at an early stage may be more productive. However, sensor level fusion is not the best choice because of the presence of noise during data acquisition. Since feature representation shows much more relevant information corresponding to the class, the fusion at feature level is expected to provide better results. Thus, we fused data at feature level, in order to provide maximum relevant information to our recognition system. We fused the feature vector of each sensor with the touch-type feature vector, making a feature vector of 30 features. The reason for fusing only two sensors is to prevent over-fitting. Larger feature vectors may end up with over-fitting of the classifier.

3.4.4 Analysis

We used the WEKA Experimenter Workbench for the classification of these patterns. Data files were converted to Attribute Relation File Format (ARFF) files and later these ARFF files and two classifiers were added to the WEKA Experimenter Workbench. We collected 30 observations for each activity from each user. We performed stratified cross-validation for training and testing of both classifiers, because of equal patterns representation from each class assuming that it will arrange the data such that in each fold, each class comprises around half the instances. Another reason is to test the classifiers with maximum possible user patterns.

3.5 Results

We achieved acceptable authentication rates for all the activities from individual sensors especially variants of accelerometers. As it can be very difficult to type while *walking*, *going downstairs* and *going upstairs*, we can expect a little increase in error rates in those two situations. However, *Touchstroke* performed well even in these positions, yielding acceptable authentication results (see Tables 3.1 and 3.2). We report our results in terms of TAR and FAR values only to avoid the redundancy, i.e., as $FRR = 1 - TAR$, $FAR = 1 - TRR$.

Table 3.1: BayesNET classifier results for fused data for all user positions (averaged over all 12 users).

	Sitting		Standing		Sofa		Walking		Upstairs		Downstairs	
Sensors	TAR	FAR	TAR	FAR	TAR	FAR	TAR	FAR	TAR	FAR	TAR	FAR
Raw + Touch	0.97	0.03	0.98	0.03	0.98	0.02	0.99	0.02	0.97	0.03	0.98	0.03
LPF + Touch	0.97	0.03	0.98	0.03	0.98	0.04	0.99	0.02	0.97	0.03	0.97	0.03
HPF + Touch	0.94	0.06	0.97	0.04	0.96	0.04	0.97	0.03	0.96	0.05	0.96	0.05
Grav + Touch	0.97	0.04	0.98	0.03	0.98	0.02	0.98	0.03	0.97	0.04	0.97	0.04
Gyro+Touch	0.95	0.05	0.97	0.03	0.96	0.05	0.98	0.02	0.96	0.05	0.97	0.04
Mag + Touch	0.97	0.03	0.97	0.02	0.99	0.02	0.96	0.05	0.95	0.06	0.96	0.04
Orient + Touch	0.96	0.04	0.98	0.03	0.97	0.03	0.98	0.03	0.96	0.04	0.97	0.04

Table 3.2: RF classifier results for fused data for all user positions (averaged over all 12 users).

	Sitting		Standing		Sofa		Walking		Upstairs		Downstairs	
Sensors	TAR	FAR	TAR	FAR	TAR	FAR	TAR	FAR	TAR	FAR	TAR	FAR
Raw + Touch	0.97	0.03	0.98	0.02	0.99	0.01	0.98	0.02	0.98	0.02	0.98	0.02
LPF + Touch	0.97	0.03	0.98	0.02	0.96	0.04	0.99	0.01	0.98	0.02	0.98	0.02
HPF + Touch	0.95	0.05	0.96	0.04	0.96	0.04	0.98	0.02	0.96	0.04	0.96	0.04
Grav + Touch	0.97	0.03	0.98	0.02	0.99	0.01	0.98	0.02	0.97	0.03	0.97	0.03
Gyro+Touch	0.96	0.04	0.97	0.03	0.96	0.04	0.98	0.02	0.96	0.04	0.97	0.03
Mag + Touch	0.98	0.02	0.99	0.01	0.99	0.01	0.96	0.04	0.95	0.05	0.96	0.04
Orient + Touch	0.97	0.03	0.97	0.03	0.97	0.03	0.97	0.03	0.96	0.04	0.96	0.04

The purpose of fusion of each sensor with touchstroke data is twofold. Firstly, to improve authentication accuracy; ROC curves for both the classifiers show an improvement in accuracy for fused data (see Figures 3.3b and 3.3d). Secondly, to make the system more secure; it is comparatively difficult to mimic two behaviors at the same time. Both classifiers worked well in all the activities and their corresponding ROC's are very accurate, we present ROC curves for *sitting* activity only.

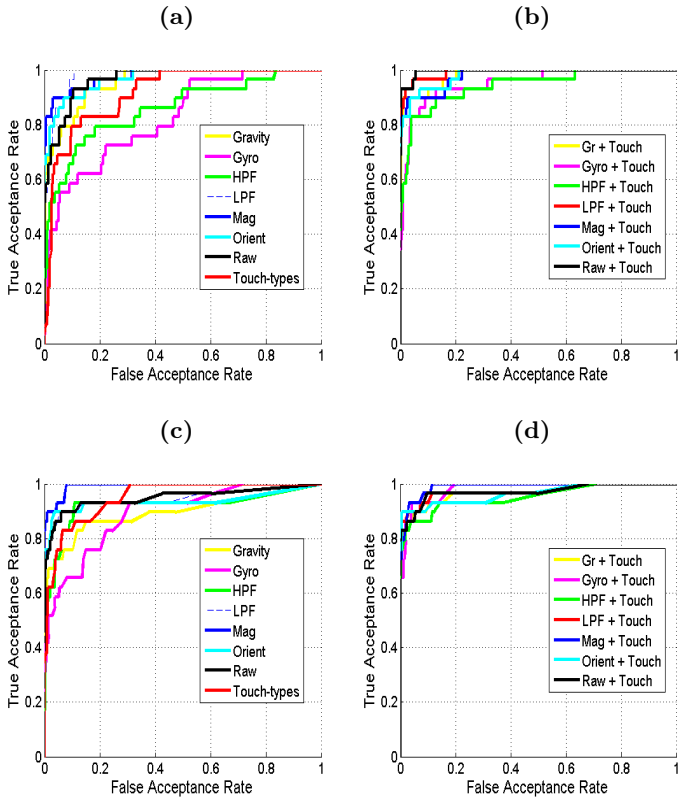
Another important observation is related to the way users hold the phone. Some users use one hand and others use both hands for holding and entering the *text-independent* text. *Touchstroke* works for both types of user. Our experiments are preliminary since we run the tests with a limited number of users who are not representative of the general population, thus we cannot exclude some bias due to the particular composition of our test set.

3.6 Chapter Summary

We propose a bi-modal biometric system, *Touchstroke*, for smartphone user authentication based on phone movement patterns and *free-text* 4-digit touch-type patterns.

The initial experiments indicate that our solution is highly accurate in each situation. Each sensor can potentially be used with touch-type features for user authentication. Our solution can be implemented in any off-the-shelf smartphone without the need for additional hardware, hence can be used as a stand-alone method or can be complemented by traditional passwords for additional security.

Figure 3.3: ROC curve for BayesNET (a) for Individual and (b) for fused sensors and RF (c) Individual and (d) fused sensors.



As the future work, we will test whether or not the fusion of multiple sensors and/or with touchstrokes has an impact on accuracy. Further, in order to check the impact of the length of the touch-type, we will investigate whether or not typing a long-digit password/PIN gives different results from those obtained for 4-*digit* entries. We will prototype *Touchstroke* with $1-class$ verification with selected features and will evaluate it in terms of performance, security, and usability, etc.

Chapter 4

Hold & Sign: A Novel Behavioral Biometrics for Smartphone User Authentication

In this chapter, we present a new bi-modal behavioral biometric solution, i.e., *Hold & Sign*, for user authentication. *Hold & Sign* takes into account the user's hand micro-movements and the user's finger movement while the user signs/writes on the touchscreen. More specifically, it profiles a user based on how she holds the phone and based on the characteristics of the points being pressed on the touchscreen, and not the produced signature image. We have implemented and evaluated our scheme on commercially available smartphones. Preliminary results with 1-class Multilayer Perceptron (MLP) prove *Hold & Sign* as an accurate, robust, power-friendly and usable, authentication solution.

Part of this chapter is published in [5]: Attaullah Buriro, Bruno Crispo, Filippo Del Frari, and Konrad Wrona. Hold & Sign: A Novel Behavioral Biometrics for Smartphone User Authentication, in proceedings of the Mobile Security Technologies (MoST) workshop held in conjunction with IEEE Security and Privacy (IEEE S&P 2016), 2016.

4.1 Introduction

A handwritten signature establishes a user's identity based on how she writes her name. This behavioral modality is very popular because it is socially and legally accepted as a means of personal identification in everyday life, however its implementations require dedicated pads [50]. Modern touchscreens make it feasible to implement handwritten signatures in smartphones and tablets.

This chapter presents a smartphone user authentication system based on how a user holds her phone while signing on its touchscreen. The system profiles pressed screen points (so-called *touch-points*) and the micro-movements of the phone during the signing

process in order to verify the user's identity.

Although typing a PIN is easier than writing something on the touchscreen, a PIN can be forgotten, whereas most users remember their own name. Moreover, launching shoulder surfing and smudge attacks to steal PINs and passwords is relatively easy. In our method, even if an attacker knows what is being written, access is still denied because he cannot mimic the phone movements of the legitimate user.

We registered the phone micro-movements using multiple physical sensors available on most smartphones. These sensors are triggered when a user starts writing (first touch-point), and stops when the user finishes writing (last touch-point). We do not take into account the signature image because it can be copied and mimicked [51]. We tested our mechanism over a dataset collected from 30 users, by applying the anomaly detection ($1 - class$) approach. Results show that using MLP as verifier, we achieve $\approx 95\%$ TAR and 3.1% FAR.

4.1.1 Contributions

The main contributions of this section are:

- The proposal and implementation of *Hold & Sign*, a new behavioral biometric user authentication mechanism, based on how the user holds her smartphone in her hand and signs her name on the smartphone touchscreen. It combines two behavioral modalities. Furthermore, it implements dynamic handwritten signature verification using multiple sensors that do not require the use of a dedicated device to capture the signature.
- Experimental validation considering how different situations, in which a user can use the device, can affect the robustness and accuracy of the biometrics.
- Performance and power consumption analysis during acquisition, training and testing phases. A preliminary usability analysis was carried out to assess how end-users reacted to our solution.

4.2 Related Work

Researchers have proposed several biometric-based solutions for smartphone user authentication. In this section, we survey the most relevant approaches.

4.2.1 Sensor-Based Authentication

Physical 3-dimensional sensors – such as accelerometers, gyroscopes, and orientation sensors – are built into most smartphones. These sensors have been used to identify users

based on their walking patterns [52], arm movements [28], arm movement and voiceprints [29], gesture models [53], and *free-text* typing patterns [27].

Li et al. [54] investigated the role of three sensors, namely the accelerometer, orientation sensor, and compass, in addition to the touch gestures in continuous user authentication. They proposed a transparent mechanism, which profiles finger movements and interprets the sensed data as different gestures. It then trains the Support Vector Machine (SVM) classifier with those gestures and performs authentication tasks. The authors achieved 95.78% gesture recognition accuracy on a database of 75 users.

Zhu et al. [53] proposed a mobile framework, *Sensec*, which makes use of sensory data from the accelerometer, orientation sensor, gyroscope, and magnetometer and constructs a user gesture model of the phone usage. Based on this gesture model, *Sensec* continuously computes the sureness score, and authorizes the real users to enable/disable certain features to protect their privacy. Users were asked to follow a script, i.e. a sequence of actions; the sensory data was collected during the entire user interaction. *Sensec* identified a valid user with 75% accuracy and it detected an adversary with an accuracy of 71.3% (with 13.1% FAR) based on 20 recruited users.

Our earlier approach *Touchstroke* [27] authenticate users using a sensor-enhanced touch stroke mechanism based on two human behaviors: how a user holds her phone and how she types her 4-digit *free-text* PIN. Using a BayesNET classifier and a RF classifier, we achieved 1% EER.

A recent study [31] makes use of Hand Movement, Orientation, and Grasp (HMOG) to continuously authenticate smartphone users. HMOG transparently collects data from the accelerometer, gyroscope, and magnetometer when a user grasps, holds and taps on the smartphone screen. On a dataset of 100 test subjects (53 male and 47 female), HMOG achieved the lowest EER of 6.92% in *walking* state with an SVM verifier.

All the solutions given above use some of the 3-dimensional sensors available in most of the smartphones and confirm the potential of these sensors for user authentication. Our solution uses 3-dimensional built-in sensors in combination with handwritten signatures to achieve a high accuracy for authentication.

4.2.2 Touch-Based Authentication

User authentication based on touch-interaction is a comparatively less explored area. Touch-interactions can be used both for one-shot login and continuous user authentication [55]. Touch-based features may include time, position, the size of touch, pressure and touch velocity, etc. De Luca et al. [56] profile touch data generated during different slide operations for unlocking the smartphone screen. Using the DTW algorithm, they achieve 77% authentication accuracy.

Angulo et al. [57] suggest an improvement to the phone lock patterns. Their system

authenticates users based on the lock patterns combined with the touch data associated with those lock patterns. They try multiple classifiers and they achieve an EER of 10.39% using a RF classifier.

Sae-Bae et al. [58] use specific five-finger touch gestures. They achieve an accuracy of 90% on the Apple iPad. However, the method is not feasible for the small touchscreens of typical smartphones. Shahzad et al. [59] consider customized slide-based gestures to authenticate a smartphone's users. Their study yielded an EER of 0.5% with the combination of just three slide movements. Sun et al. [60] require users to draw an arbitrary pattern with their fingers in a specific region of the screen for unlocking their smartphones. Users were authenticated on the basis of geometric features extracted from their drawn curves along with their behavioral and physiological modalities. The solution presented in [61] by Sae-Bae and Memon is conceptually similar to our work. This unimodal online signature verification scheme extracts the histogram features from the user signature and performs user authentication. The lowest EER achieved was 5.34% across different sessions.

Our solution relies on the screen touch-points being pressed and the velocity of finger movement during the signing – neither signature image nor its geometry is used. It does not require the user to draw specific patterns for authentication, but simply use any pattern, which is convenient or well-known to her - e.g. to sign her name. This increases usability of our solution as the user is not required to perform an initial learning of an unknown pattern in order to memorize it and for his signing features to become stable and reliable.

4.2.3 Signature-Based Authentication

Some work has been done regarding signature-based biometric authentication on smartphones [62, 63, 64]. Koreman and Morris [65] propose a continuous authentication method based on multiple modalities, namely the face, voice, and signature on the touchscreen. Their study yielded an EER of 2.3%, 17%, 4.3% and 0.6% for voice, face, signature and fused modalities respectively.

Vahab et al. [66] implement online signature verification using an MLP classifier on a subset of Principal Component Analysis (PCA) features. The validation was performed using 4000 signature samples from the SIGMA database [67] and yielded an FAR of 7.4% and an FRR of 6.4%.

In recent work of Xu et al. [68], users were asked to write different alphabetic characters on the screen; 42 handwritten features were extracted using a handwriting forensics approach (which focuses on the geometry of writing [69]). Those features were then classified using SVM. The proposed solution achieved an EER of 5.62%. Additionally, the touch slide (touch-points stimulated when writing an alphabet) yielded an EER of 0.75%.

Images of handwriting signatures have been used by SignEasy as an authentication method in iOS8[70], allowing users to transparently add their electronic signatures on important documents. Similarly, a signature recognition system [71] performs user identification based on user signatures captured via a smartphone touchscreen or via a dedicated signature capturing device. It verifies signatures by computing the similarity score between the query signature and the stored signature template. Additionally, this system provides client-server solutions based on signature images. None of them uses phone movements and/or touch features for user authentication.

Our solution is different because it is bi-modal thus intuitively more secure than the uni-modal ones; it takes into account phone movements *and* finger movements during the signing process. Spoofing only one of the two modalities would not suffice to grant access to the phone.

4.3 Approach

In this section, we illustrate the main approach adopted by our solution.

4.3.1 Intuition Assessment

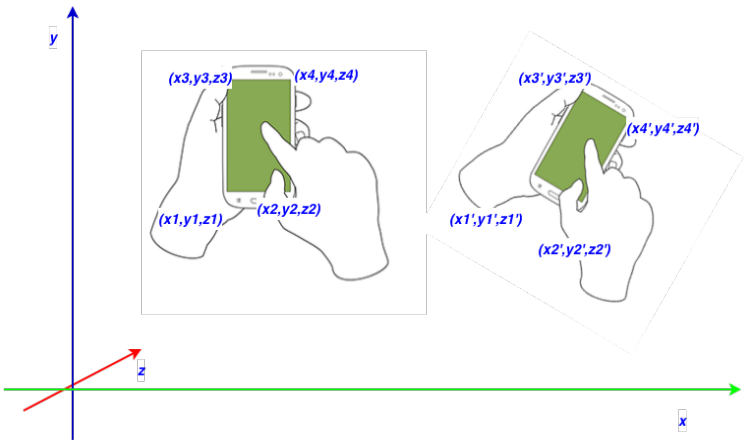
Our initial intuition is that each person holds and moves her phone in a unique way, thus generating a unique movement pattern. Due to the uniqueness of such behavior, it becomes very challenging for others to generate exactly the same movement pattern. Even in case of a successful mimicry, the movement pattern will still be different due to the differences in the structure of human body (e.g., the height and exact orientation of the phone, etc.). We showed earlier (Chapter 3 Figure 3.2) that phone movements are sufficiently discriminatory across users.

Similarly every user has a unique way of writing. An adversary can spoof and copy what is being written or how is being written, but it is very difficult to copy the exact touch locations, velocity of slide, force of touch and other touch related features.

4.3.2 Our Solution

Our solution (see Figure 4.2) exploits the phone movements in hand and finger movements on the touchscreen as shown in Figure 4.1. In particular, we consider all the touch-points pushed for the entire signature and the velocity of the finger movement. All the physical sensors are triggered and kept running during the whole signing process (from first to last touch-point) on the touchscreen. Obtained sensor readings are then preprocessed to extract useful features. As we propose a bi-modal system, we need to combine the extracted features from both built-in sensors and the touchscreen to profile user behavior. Our model involves feature selection, which entails selecting the subset of productive

Figure 4.1: Different phone positions during signing process.



features to be used for user authentication. A user profile template is formed based on the selected feature subset and is then stored in the main database. These behavioral vectors are later matched with the vector of the test sample in order to authenticate/reject the claimant.

4.3.3 Considered Sensors and Classifiers

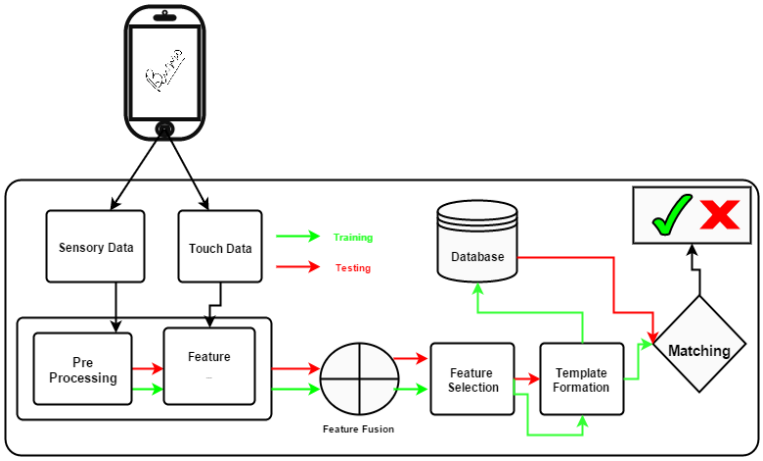
Our solution relies on three built-in 3-dimensional sensors: the accelerometer, the gravity sensor and the magnetometer, and the touchscreen.

All the above sensors generate continuous streams in X, Y and Z directions.We have added a fourth dimension to all of these sensors and name it *magnitude* as in the Chapter 3 (Section 3.3.3).

We chose four different 1-class verifiers, i.e. BayesNET, K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP) and RF, because they were found to be very effective in previous studies. BayesNET and RF verifiers were used with their default settings. However, the parameters of both MLP and KNN were optimized, because with default parameters they performed quite poorly. We used K = 3 in KNN and similarly used 3 hidden layers in MLP. We used all of our verifiers wrapped into WEKA's metaclass classifier; the OneClassClassifier.[1]

[1]http://weka.sourceforge.net/doc.packages/oneClassClassifier/weka/classifiers/meta/OneClassClassifier.html

Figure 4.2: Our proposed authentication system.



4.4 Experimental Analysis

4.4.1 Data collection

We recruited 30 volunteers (22 male and 8 female); the majority of them are either Master's or PhD. students but not security experts. In order to have diversity, we recruited users from several nationalities. The purpose of the experiment and the description of our proposed solution was clearly explained to each user individually. The process of data collection and how data are stored were carefully explained. Each volunteer provided explicit consent to participate in the experiment. We collected data in three different activities, *sitting*, *standing* and *walking* with *Google Nexus* 5 using *SENSOR_DELAY_GAME*.

4.4.2 Features

We gathered 4 data streams from every 3-dimensional sensor except touchscreen, and we extracted 4 statistical features, namely mean, standard deviation, skewness, and kurtosis, from every data stream. Data from every sensor was transformed into a 4 by 4 features matrix. In total, we obtained 16 features from all four dimensions of each sensor. Similarly, we extracted 13 features from touchscreen data. The extracted features from touchscreen data are shown in Table 4.1.

4.4.3 Feature Fusion

The extracted feature set from the data from multiple sources can be combined to form a new feature set. We used fusion at the feature level (like in Chapter 3 Section 3.4.3), in order to provide the maximum amount of relevant information to our recognition system. The fusion of 16 features from each sensor makes a new feature vector which here is

Table 4.1: List of selected features from touchscreen data.

No.	Touch Features						
1 - 7	StartX	EndX	StartY	EndY	AvgXVelocity	AvgYVelocity	MaxXVelocity
2 -13	MaxYVelocity	STDX	STDY	DiffX	DiffY	EUDistance	-

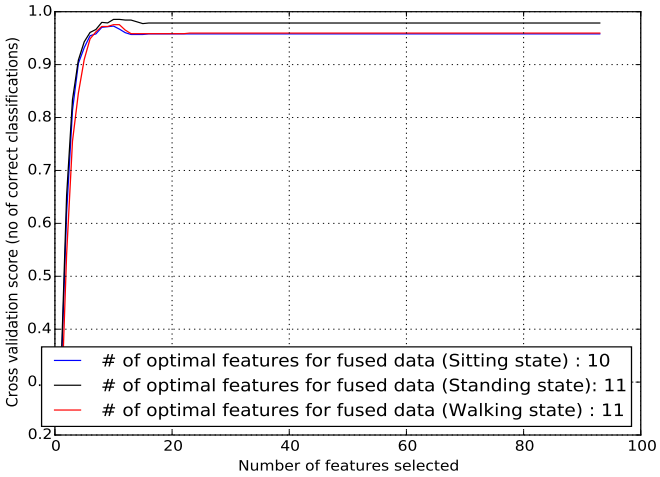
referred to as the pattern of the user's hold behavior. The length of this feature vector is 80 features (16 for each of the five used sensors). Similarly, the feature vector of sign behavior is small (13 features, extracted from the captured touch-points through the touchscreen) and we call it a sign pattern. The length of the fused feature vector for both modalities becomes 93 features.

4.4.4 Feature Subset Selection

Feature subset selection is the process of choosing the best possible subset, i.e. the set that gives the maximum accuracy, from the original feature set. Note that even if we achieve the same accuracy with reduced features, smaller feature vectors decrease computation time and allow the classifier to decide faster.

We evaluated our feature set (93 features for fused behaviors) with Recursive Feature Elimination (RFE) feature subset selection methods. We relied on scikit-learn[2], a Python-based tool for data mining and analysis, for RFE feature subset selection.

Figure 4.3: RFE Feature Selection from *sitting*, *standing* and *walking* states.



The RFE classifier trains itself on the initial set of features and assigns weights to each of them. The features with smallest weights are later pruned from the current feature set. The procedure is repeated until the intended number of features is eventually reached[3]. We applied RFE with 10-fold stratified cross-validation using an SVM classifier on the data of all activities for two classes. The plot (see Figure 4.3) shows the optimal number (11) of features selected from fused data in standing and walking state and 10 for sitting

[2]http://scikit-learn.org/stable/

[3]http://scikit-learn.org/stable/modules/feature_selection.html

Table 4.2: List of selected features from fused (bi-modal) data.

Sitting	*Standing*	*Walking*	*Combined*
MgX_Mean	HPFMag_Kurt	EndY	HPFY_Mean
RAWY_STD	RAWY_STD	RAWY_STD	HPFZ_Mean
DiffX	DiffX	DiffX	GrZ_Skew
StartY	StartX	RAWZ_Mean	StartX
MgY_Mean	EU_Distance	STDY	EndX
StartX	RAWMag_STD	HPFZ_Mean	StartY
EndY	StartY	StartY	EndY
MgMag_Mean	DiffY	HPFX_Skew	MaxYVelocity
GrY_Mean	HPFX_Mean	HPFX_Mean	AvgXVelocity
STDX	MgMag_Mean	DiffY	STDY
-	EndX	HPFY_Mean	DiffX

state.

4.4.5 Analysis

We analyzed data in two settings, i.e. (i) a *verifying legitimate user* scenario, and (ii) an *attack* scenario.

In the *verifying legitimate user* scenario, we train the system with the data from the *owner* class and then test the system with the patterns belonging to that class. The outcome can be either accept or reject. We used a 10-fold stratified cross-validation method for testing. In cross-validation, the dataset is randomized and then split into k (here $k = 10$) folds of equal size. In each iteration, one fold is used for testing, and the other $k - 1$ folds are used for training the classifier. The test results are averaged over all folds, which give the cross-validation estimate of the accuracy. This method is useful in dealing with small datasets. Using cross-validation we tested each available sample in our dataset. We report the results of these settings in terms of TAR and FRR.

In the *attack* scenario, we train the system with all the data samples from the *owner* class and then test the system with the patterns belonging to all the remaining classes (29 users). The outcome can be either false accept or true reject. We report the results of these settings in terms of FAR and TRR.

4.5 Results

We report our results in three ways: intra-activity, inter-activity and activity fusion. By intra-activity, we mean training and testing each single activity (i.e. training walking to test walking only). Inter-activity means training with one single activity and using that training for testing all activities. We tested the training for each activity. In activity fusion, we used the combined data of all 3 activities for both training and testing (i.e. training with fused data from walking, sitting and standing) to test all activities. The reason for this is that we want to check whether training in a single activity is sufficient to recognize all the testing samples across activities. Otherwise, we would need to train the recognition system with patterns of multiple activities. As the MLP verifier has consistently out-performed all other verifiers in all three activities (see Table 4.3), we will take into account only this verifier in further analysis.

Table 4.3: Results of different classifiers (averaged over all 30 users) in different activities.

	Sitting		*Standing*		*Walking*	
Classifiers	TAR	FAR	TAR	FAR	TAR	FAR
BN	0.758	0.001	0.740	0.003	0.710	0.000
MLP	0.797	0.001	0.790	0.004	0.790	0.000
IBk	0.761	0.001	0.750	0.002	0.720	0.000
RF	0.767	0.001	0.750	0.002	0.710	0.000

The results of all settings are presented below:

4.5.1 Intra-Activity

The results of all three activities, prior to feature selection (averaged over 30 users), are given in Table 4.3. We achieved $\geq 79\%$ TAR with full features in all the activities using the MLP verifier. We then applied a feature subset selection method (RFE) on our dataset. Figure 4.4 shows that we improved our authentication results (from $\geq 79\%$ to 85.56% in *sitting*, 86.75% in *standing* and 86% in *walking*) with our chosen RFE feature subsets (see Table 4.2). We obtained 85.5% to 86.7% TAR with the MLP verifier in the three user activities. In related work, [31] reported 93.08% TAR but at the expense of 6.92% FAR using the 1-class SVM verifier and [29] reported 10.28% FAR and 3.93% FRR with 1-class RF verifier.

4.5.2 Inter-Activity

In order to validate the applicability of our mechanism in multiple user positions, we tested its performance across multiple activities. For example, if we train the system with the training patterns of just the *sitting* activity and test it with the patterns of both *standing* and *walking* activities and vice versa, we can observe whether or not training with a single activity is sufficient.

Figure 4.4: Comparison of TAR for Full and RFE based feature subsets in *Intra-activity*.

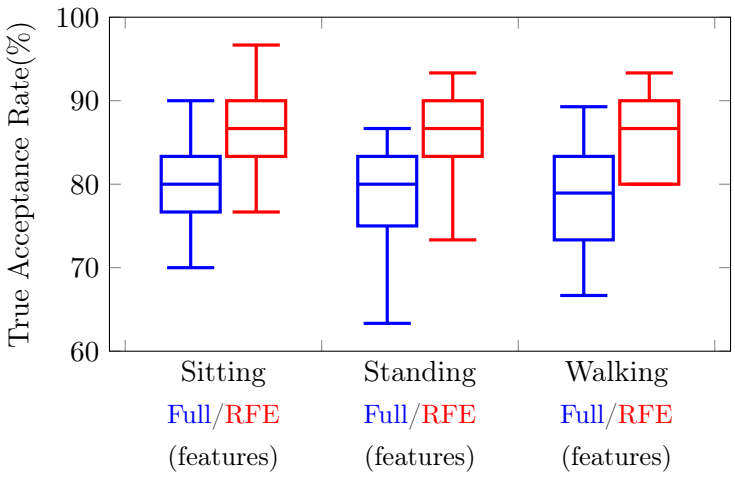


Figure 4.5 shows unsatisfactory results (65.82% at best), and thus we conclude that we need to train our system in multiple situations to increase its accuracy.

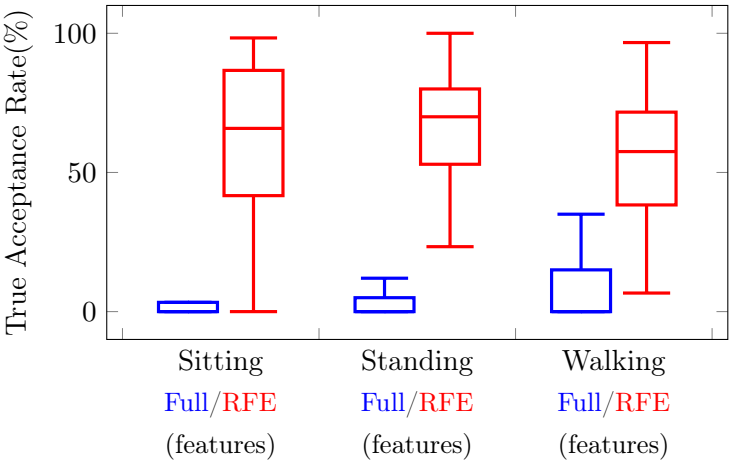
4.5.3 Activity Fusion

Training the system in just one activity and using it in multiple activities does not lead to good results. As a solution, we combined the patterns of multiple activities and applied the RFE feature selection method on the combined data. As done earlier, we picked 11 highly ranked features (see the last column of Table 4.2) and proceeded to further analysis. We applied the same methodology (as per section 4.4.5) to test our combined dataset from all three activities. The results are summarized in Table 4.4. The system achieved $\approx 95\%$ TAR at the expense of just 3.1% FAR. We observed that activity fusion could be useful in terms of usability (as it requires one-time training in multiple activities) and accuracy (we obtained $\approx 95\%$ TAR) so we checked its efficacy with the final implementation of *Hold & Sign*. We trained the system with a different set of training patterns from different activities and used the same set of features (see the last column of Table 4.2) and compared the results.

Table 4.4: Results of MLP (averaged over all 30 users) for combined data of all three activities.

	Combined data from all activities			
Classifiers	TAR	FRR	FAR	TRR
MLP	94.8	5.2	3.1	96.9

Figure 4.5: Comparison of TAR for Full and RFE based feature subsets in *Inter-activity*.



4.6 Hold & Sign Implementation

We developed the final prototype of *Hold & Sign* taking into consideration all our findings. *Hold & Sign* uses the MLP classifier based on the feature set extracted using the RFE method. The analysis was performed using this application on a Google Nexus 5 smartphone running Android `4.4.4`. Screenshots for training and testing are shown in Figure 4.6. *Hold & Sign* requires a minimal configuration, i.e. a user may choose either both modalities or any one of them (as shown in Figure 4.6b) and needs to train the classifier accordingly. The user can also decide the number of training instances, i.e. how many times to write his own name on the touchscreen to train the classifier (Figure 4.6c). In all choices, the user is helped by the display of suggested recommended values. The user is later required to write his own name for authentication (see Figure 4.6d).

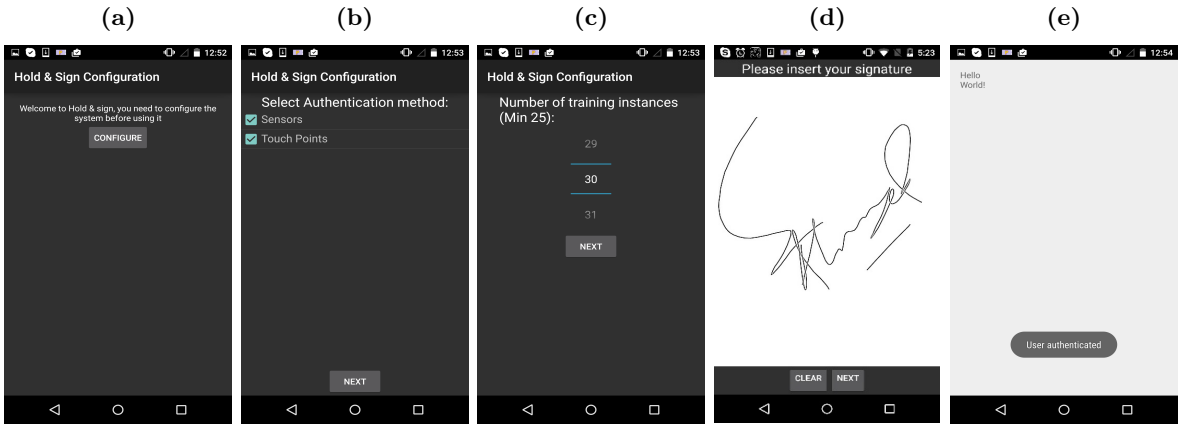
4.6.1 Performance

We tested the performance of *Hold & Sign*. We measured three different timings: sample acquisition time, training time and testing time. We computed these times for 3 different settings: with 15, 30 and 45 patterns. We tested each setting on the Google Nexus 5 with 35 tries for each time. Results are averaged over all 35 runs.

4.6.1.1 Sample Acquisition Time

This is the time used by the user to provide a sample for authentication. It is important to know it because users may feel annoyed by the required acquisition time that possibly results in complete removal of the *Hold & Sign* application. We compared the sample

Figure 4.6: Screenshots of *Hold & Sign* in training (a to d) and testing phase (d & e).



acquisition time for multiple mechanisms in Table 4.5. What makes our acquisition fast is the free-text feature, e.g. the user can write any word (e.g., her own name).

4.6.1.2 Training/Testing Time

Training time is the time required to train the classifier. It is usually computed just once, at the installation, when the training samples are provided to the system. In contrast, testing time is the time required by the system to accept/reject the authentication attempt. Our mechanism took 3.497s, 6.193s and 9.310s for classifier training with 15, 30 and 45 patterns, respectively. Similarly, the testing times with 15, 30 and 45 patterns were $0.200s$, $0.213s$, and $0.253s$, respectively. Comparison with the performance of other recent proposals is shown in Table 4.6.

4.6.2 Power Consumption

Generally, it is quite difficult to determine with high accuracy the power consumption of a single mobile application. Using dedicated hardware allows high accuracy [31]. However, there are software-based approaches that though less accurate, are being extensively used [72]. Since we wanted an initial indication, we used the software-based approach.

In order to check the overhead resulting from use of the application (in different steps), we terminated all the running applications and all Google services, switched off WiFi, Bluetooth, and cellular radios. The screen was kept running for the entire duration of the experiment with brightness at the lowest level and automatic brightness adjustment

Table 4.5: Sample acquisition time for different methods adapted from [1].

Method	Sample Acquisition Time (s)
Our method	3.5
PIN	3.7
Password	7.46
Voice	5.15
Face	5.55
Gesture	8.10
Face + Voice	7.63
Gesture + Voice	9.91

disabled. A similar approach is applied in[72]. We used *Trepn*[4] and performed the experiments as follows:

In the first step, we computed reference power consumption by running *Hold & Sign* with all the steps (sensor data collection, feature extraction, etc.) disabled. In the second stage, we enabled the sensor data collection part only to compute the overhead resulting from sensory data collection. In the third stage, we enabled the feature extraction part to compute the power consumption resulting from this process. In the final step, we analyzed the app with all its functionalities. We profiled the power consumption for all these settings of *Hold & Sign* for the entire duration of the experiment (shortest duration 1 minute and 50s and longest 2 minutes and 40s) with 35 attempts each. The reference power consumption is $460mW$. We observed a 7.17% overhead ($493mW$) for sensor data collection, 27.8% in both data collection and feature extraction stages ($588mW$) and $\approx 1000mW$ in all stages of the final setting. The feature computation incurred just a 19.2% overhead corresponding to data collection.

We observed that the average power consumption of our mechanism is very low, which makes it a power-friendly app. This claim can be supported by looking at some common smartphone tasks and their average power consumption [73][74]:

- A one-minute phone call: $1054mW$
- Sending a text message: $302mW$
- Sending or receiving an email over WiFi: $432mW$
- Sending or receiving an email over a mobile network: $610mW$

[4]https://play.google.com/store/apps/details?id=com.quicinc.trepn&hl=en.

Table 4.6: Comparison of our results with state of the art.

Ref.	Devices	Classifier	No. of Users	Training Time	Testing Time
Our method	Nexus 5	MLP	30	3.5 - 9.3s	0.215 - 0.250 s
Lee et al. [75]	Nexus 5	SVM	8	6.07s	20s
Li et al. [54]	Motorolla Droid	Sliding Patterns	75	n.a	0.648s
Nickel et al.[76]	Motorolla Milestoon	KNN	36	90s	30s

4.7 Usability Analysis

We report the usability of our mechanism in two ways: based on how many patterns are enough for training the classifier to achieve significant authentication accuracy, and by applying the standard System Usability Scale (SUS) for collecting users' views about our proposed mechanism.

4.7.1 Tradeoffs between Training and Accuracy

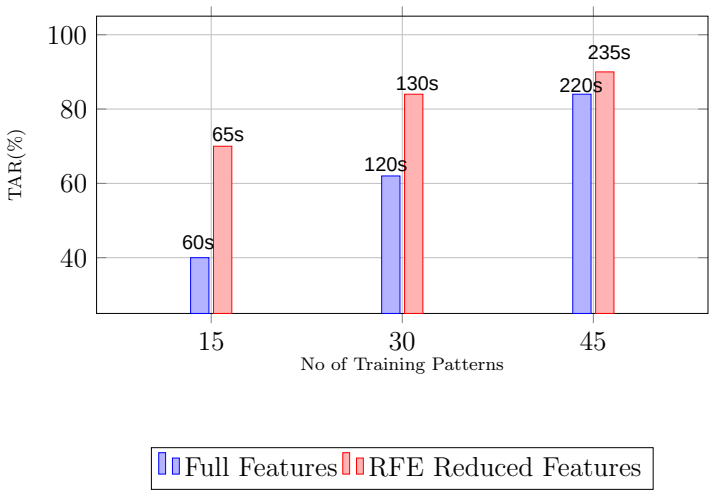
As shown in Table 4.5, the average duration of a signature drawn by a user on the touchscreen was 3.5s with the lowest value being 2s. In our test, we observed that the willingness of users to participate in our testing is strongly related to the amount of time spent for training. We expect a similar dependency also in normal usage. Hence it is important to evaluate the ratio of training time to accuracy. We observed that with just 15 patterns (in which case a user may take less than a minute to train the system), the user could be identified with around 70% TAR. Accuracy can be increased at the cost of training time. It took less than 4 minutes for the slowest of our testers to train the system with 45 patterns (15 in each activity) and authentication results were $\approx 90\%$. The TAR percents are averaged over 35 user attempts. The results are shown in Figure 4.7.

4.7.2 Evaluation

We distributed *Hold & Sign* along with an 11-question questionnaire adapted from the System Usability Scale[5] (SUS) to our chosen volunteers (30 users). The SUS assessment tool is widely used for gathering subjective impressions about the usability of a system. It has already been used in the context of smartphone authentication [1]. The response to each question can be given on a five-point scale ranging from 'Strongly Disagree' to 'Strongly Agree'. The SUS score is a value between 0 and 100 where a higher value indicates a more usable mechanism. A raw SUS score can be transformed to a percentile [77] or to a grading scale[78], allowing an easier interpretation of results. The average SUS

[5]http://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

Figure 4.7: User authentication on the prototype application. This figure verifies the average results obtained from the fusion of activities as described in Section 4.5.3.The values above the bars indicate time spent to provide samples



score is 68. Like the previous study [1], we added a question to this questionnaire: *What did you like or dislike about the mechanism?* This question was optional and subjective; users were suppose to write some lines supporting the reason(s) for liking or disliking our mechanism. We wanted to collect early feedback to allow us to improve our solution in future.

We asked the users to use our app for some days (preferably a week) and share their experience with us. We received responses from 18 out of 30 volunteers (60%).

4.7.3 Responses

We received useful feedback on our mechanism. We achieved an average SUS score of 68.33%. Our score is better than the well-established voice recognition score (66%) and its fusion with the face (46%) and gestures (50%) as reported in the literature [1]. Most of the responses were positive about the use of signing as an authentication credential. Most of the participants were also positive and comfortable using a finger and using the smartphone touchscreen (i.e., no complaints about the size of the display). We also got some negative responses, mostly related to the initial setup; it was “too cumbersome" for some, i.e. “a user has to sign multiple times in order to train the system whereas setting up a PIN is easier". We also received some negative responses regarding the system requiring the use of both hands.

Our mechanism is clearly in the initial stages and requires more tuning for increasing

its user acceptability. We are planning to incorporate these initial suggestions into future versions of *Hold & Sign* and also to run more extensive the usability studies.

4.8 Limitations

Our current solution suffers from two important limitations. Firstly, also pointed out by a volunteer, users must use both hands. One hand holds the phone and other hand's fingertip is used for the signature. The user, therefore, may experience some difficulty in using our solution, especially when on the move. Secondly, the system cannot predict the user's ongoing activity in order to extract the best pre-selected features and use them for verifying user identity.

4.9 Chapter Summary

We proposed a new bi-modal behavioral biometric authentication solution, *Hold & Sign*, using as behaviors how a user holds her phone and how she writes on the touchscreen. We achieved 79% TAR at "*zero*" FAR from 1-class MLP with full features in *walking* activity. The reason for this achievement could be the fact that during *walking*, sensors gather more data thus is possible to build accurate patterns. After applying feature subset selection, TAR improved to 86.7% at the expense of just 0.1% FAR. Lastly, processing the data from combined activities yielded 94.8% TAR at 3.1% FAR.

Hold & Sign requires on average just 3.5s to enter the behavioral pattern. Its ability to authenticate/reject a user within $0.215 - 0.250$s makes it very fast. The closest reported testing time in the literature is 0.648s [54].

Hold & Sign offers two advantages over traditional mechanisms. Firstly, a user can write his own name in an unconstrained way with a finger on the smartphone's touchscreen, which makes memorability and repetition easier.

There is no need to remember a password/pattern and no need to keep them secret, thus eliminating the problem of sharing and stolen passwords. Also, it is easy to integrate and implement in most modern smartphones without the need for additional hardware. *Hold & Sign* can be used as a stand-alone method or can be used in conjunction with other well-established mechanisms for additional security.

Since signature-based authentication is already deployed for user identification and it is also very common to use finger movements for navigating documents, e.g. web pages, photo albums, messages, etc., we expect our solution to receive wider user acceptance. The results of the preliminary usability analysis, with a SUS score above the average (68.33%), indicate a positive starting point.

As a future work, we plan to investigate the permanency of this biometric modality, extend our work in terms of continuous authentication and explore its usability with

a larger and more heterogeneous sample of testers. We are also going to address the problem of seamless and fast detection of a user's current activity since this would allow authenticating users based on the best feature subset selected from that particular activity.

Chapter 5

Please Hold On: Unobtrusive User Authentication using Smartphone's built-in Sensors

In this chapter, we propose a novel method of fully unobtrusive user authentication. Our approach is based on profiling the user's hand micro-movements after an unlock event occurs. Generally, it requires a slide-to-unlock or PIN, password or pattern to unlock the smartphone. In any case, whenever the user performs any of these actions, Android operating system sends a special broadcast event - the `USER_PRESENT`. Our proposed method exploits the user's hand movements for a limited time after the event occurs and ensures authenticity for every session. Extensive experimentation with multiple machine learning classifiers proves the efficacy of our mechanism. We report an authentication accuracy of 96% with an EER of just 4%. Our proposed method can be used as a standalone solution or can be complemented with any of the existing authentication mechanism to improve the authentication accuracy and robustness.

The part of this chapter was accepted for publication in [79]: Attaullah Buriro, Bruno Crispo, and Yury Zhauniarovich, Please Hold On: Unobtrusive User Authentication using Smartphone's Built-in Sensors, in proceedings of the IEEE International Conference on Identity, Security and Behavior Analysis 2017 (ISBA-2017), New Delhi, India, February 22-24, 2017.

5.1 Introduction

This chapter presents a novel approach for *unobtrusive user authentication* based on profiling of hand micro-movements after the unlock event. Our system is activated after a user unlocks a phone and collects interaction patterns using built-in *unprivileged sensors* within a short period of time. Afterwards, using machine learning approach it assesses

if the smartphone has been activated by the owner or an impostor. Our system does not require to present any token or to perform some remembered actions. Thus, it is completely transparent and is applicable both for smartphones with and without authentication mechanisms enabled. It uses only unprivileged sensors so it does not require any permission, interaction or cooperation.

Our experimental evaluation confirms the practicality of our approach. Indeed, in the authentication task we managed to achieve an accuracy of 96% at an EER of 4%. Our system can be used as a standalone authentication mechanism and in multi-modal approaches as the ones proposed in [80]. Since the data collection and user authentication is performed in the background, we claim that our method is fully unobtrusive, thus, has a wide user acceptability.

5.1.1 Contributions

The main contribution concerning this chapter are listed below:

- The proposal of a novel approach for fully transparent user authentication on mobile devices using built-in unprivileged sensors.
- The validation of the approach on a dataset collected from 53 users. The dataset consists of readings collected from multiple sensors, user actions data and smartphone model information.
- Assessment and evaluation of different time periods needed to collect training samples and the amount of data required to be supplied to the authentication system.

5.2 Related Work

In this section, we survey the sensory-data-based authentication schemes for smartphone user authentication proposed over the years.

Shi et al. [81] presented a multi-sensor approach to passively identify a genuine user. Their system uses an accelerometer, touch screen, voice and location data for user authentication. They reported $\sim 97\%$ TAR, using the Naive Bayes classifier, on a dataset of 7 users (three females and four males). Li et al., [54] explored the utility of three different sensors: the accelerometer, the orientation sensor, and compass, in addition to the touch gestures for continuous user authentication. Their method profiles finger movements using classical touch-based features and interprets the sensed data as different gestures. An SVM classifier is then trained with gestures to perform authentication tasks. Accuracy of 95.78% is reported on a database of 75 users.

Zhu et al. [53] propose a mobile framework model *Sensec* based on an accelerometer, the orientation, a gyroscope, and a magnetometer to construct a user gesture profile.

The model then continuously computes the sureness score to authenticate the user. By concatenating X, Y, Z values from the aforementioned sensors, a valid user is identified with 75% accuracy and an adversary with an accuracy of 71.3% (with 13.1% FAR) from a set of 20 users. However, the study requires a user to follow a fixed protocol and collects data for the entire user interaction session. The method proposed here is different since it does not require any specific protocol to be followed. Furthermore, data is collected only once in the entire session (without requiring any explicit user interaction).

Conti et al. [28] exploit accelerometer and orientation sensor readings collected during call placing/answering, to profile the genuine user. Their study reports an FAR of 4.44% at an FRR of 9.33% on a dataset of 10 users using DTW as the classifier. Later, we extended this work ([29] see Chapter 7) to a tri-modal system which involves arm movement, finger swiping and voice recognition. We reported a 10.28% FAR at a 3.93% FRR on a dataset of 26 users. An important related work, i.e., HMOG by Sitova et al. [31] leverages *Hand Movement, Orientation, and Grasp* to continuously authenticate smartphone users. It transparently collects data from the accelerometer, gyroscope, and magnetometer when a user grasps, holds and taps on the smartphone screen. On a dataset of 100 test subjects (53 males and 47 females), HMOG achieves the lowest EER of 6.92% in *walking* state with SVM classifier. Our presented method does not require any typing, keystrokes or grasp. Instead, the data is collected transparently after an unlock event occurs (as a result of either slide-to-unlock, entering PIN or password, etc.).

Google project - ABACUS, built a large dataset containing 27.62 TB of smartphone signals on Nexus 5 smartphones from 1500 users over a period of six months [82]. The data was obtained from multiple sensors, namely, camera, touchscreen, keyboard, accelerometer, magnetometer, gyroscope, light sensor, GPS, Bluetooth, Wi-Fi and application usage. The data was recorded for the entire user interaction session - from one smartphone unlock to the next time it is locked. Using optimized shift-invariant Dense Convolutional Mechanism (DCWRNN) an EER of 8.82% (per session) and 15.84% (per device) was reported. Here an EER of 8.82% means that 91.18% of the times, the correct user was holding and moving the phone, not necessarily interacting with it. In our case, we identify the user after her interaction with the device. Upal et. al., [83], collected smartphone signals from 48 volunteers on a Nexus 5 smartphone, over a period of two months. They collected data from the camera, touchscreen, gyroscope, accelerometer, magnetometer, light sensor, GPS, Bluetooth, WiFi, proximity sensor, temperature sensor and pressure sensor. Apart from face detection and recognition results, they reported swipe-based authentication results. Among multiple classifiers, the Random Forest classifier achieved the lowest EER of 22.1%. However, both datasets have not yet been made available to the research community, hence it is difficult to have a direct comparison to these solutions.

Most sensor-based authentication solutions listed above utilize the sensor(s) available

in smartphones. They collect sensory data associated with either finger movements, user tappings or associated with the particular motion (e.g., call placing). Furthermore, most solutions are based on the data collected in laboratory settings. On the other hand, our method is different in the following ways:

- It is fully unobtrusive. It does not require any permission, participation, or cooperation from a user. Each authentication step is performed, transparently, in the background.
- Data was collected in a totally uncontrolled manner.
- Our method utilizes all the 3-dimensional sensors available on the smartphones.
- Our scheme initiates all the sensors after receiving the user presence notification from the OS associated with the *USER_PRESENT* broadcast receiver. Therefore, it can complement the existing one-shot login methods and becomes more useful, especially, for those users (e.g., slide-unlock users) who do not want to invoke any explicit authentication mechanisms on their smartphones.

5.3 Approach

5.3.1 Intuition Assessment

It has been reported in previous studies [27, 5, 31, 28, 82] that each user holds, interacts and moves her phone in a unique way (see graphs presented in Chapter 3 (Section 3.3.3)). This uniqueness of movement pattern increases the authentication accuracy on the one side and makes it challenging for the impostors to exactly generate the same movement patterns on the other.

5.3.2 Our Solution

Our proposed method is based on the idea of utilizing the user's hand micro-movements after she unlocks her phone using an authentication method, e.g., PIN, slide-to-unlock, etc. In either case, when the user unlocks her smartphone, the Android OS generates a specific broadcast event *USER_PRESENT*. The mentioned event is generated only once per session (when the user unlocks her smartphone). Similar events[1] are generated also in other mobile operating systems, e.g., iOS. Thus, the proposed method can be implemented also for other popular mobile operating systems.

Our idea is based on profiling the user's hand micro-movements for a short period of time (at most 10 sec). The rationale behind choosing this time duration is the following:

[1] e.g., `PhoneApplicationFrame.Unobscured` event in Windows Phone OS, or `com.apple.springboard.lockstate` event in Apple iOS.

(i) it was empirically determined that this time is sufficient enough for pattern discrimination, and (ii) this duration is too short for an adversary to debug the device [84]. The collected data is pre-processed and relevant features are extracted. A final template is constructed by concatenating all the extracted features, and then it is fed to the classifier for training or for testing (see Figure 5.1). If during this period a user is classified as a genuine user, the system will not interrupt the owner's interactions with the smartphone. On the other hand, if the user is classified as an impostor, the system will alert the owner of the phone (e.g., sending an email), and may stealthily isolate the impostor from accessing sensitive functionality [85, 81], or ask for explicit authentication [86, 87]. We restrict ourselves to collecting information from unprivileged sensors. This allows our system to be implemented as a separate authentication service or to be integrated within an implicit authentication framework as the one proposed in [88]. Figure 5.1 illustrates our proposed approach for user authentication on mobile devices. The sensory data is first pre-processed and the features are extracted. The extracted features are then concatenated together, to make a feature vector, and this feature vector is fed into the feature selection module to find the most productive feature subset for onward user profiling. The selected feature subset is stored in the database for matching afterwards with the query sample to accept or reject the user.

5.3.3 Considered Sensors and Classifiers

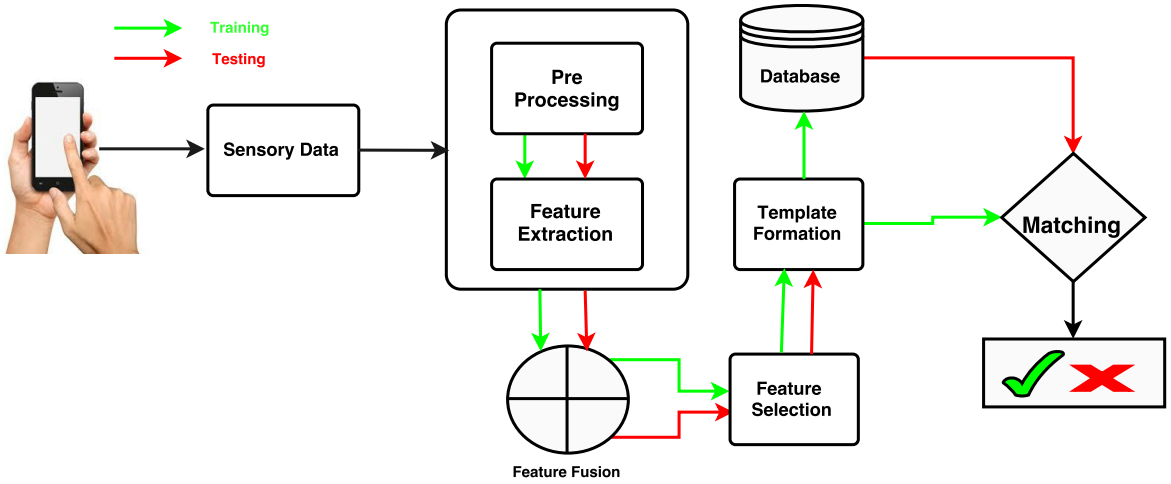
Our solution is based on collected data from multiple sensors: the accelerometer, i.e., Raw, LPF and HPF, gravity, gyroscope, magnetometer and orientation. Thus, we used in total seven sensors (5 physical and 2 logical). We calculated magnitude dimension from the 3 dimensions of each 3-dimensional sensor.

We used four classification algorithms from the WEKA workbench for user authentication: BN, KNN, MLP and RF.

Table 5.1: Dataset Description.

User ID	Manufacturer	Model	Start Version	End Version	Kernel Updated	Samples	Period	All Sensors
1	Samsung	Galaxy Nexus	4.3	4.3	54	144	12.20	52
2	QMobile	V5	4.2.1	4.2.1	54	229	198.78	52
3	Samsung	GT-I9300	4.3	4.3	54	146	9.88	52
4	Samsung	GT-I9506	4.4.2	4.4.2	52	807	17.19	52
5	Samsung	GT-S7580	4.2.2	4.2.2	54	3	0.04	54
6	LGE	Nexus 5	4.4.4	4.4.4	54	265	10.25	52
7	Samsung	Nexus S	4.1.2	4.1.2	54	2	0.05	52
8	Sony	D6603	4.4.4	4.4.4	54	937	15.29	52
9	LGE	Nexus 5	4.4.4	4.4.4	54	7	0.12	52
10	Samsung	GT-S7275R	4.2.2	4.2.2	54	124	9.37	54
11	Samsung	GT-I9506	4.4.2	4.4.2	54	397	11.13	52
12	Samsung	GT-S7580	4.2.2	4.2.2	52	1043	27.20	54
13	QMobile	A600	4.1.2	4.1.2	54	121	4.69	54
14	Samsung	GT-I9506	4.4.2	4.4.2	54	2	0.01	52
15	LGE	Nexus 4	4.4.4	4.4.4	54	741	17.76	52
16	LGE	Nexus 4	4.4.4	4.4.4	54	343	19.43	52
17	Samsung	GT-I8150	2.3.6	2.3.6	54	70	12.04	54
18	Samsung	GT-S7562	4.0.4	4.0.4	54	410	14.76	54
19	Samsung	Nexus S	4.1.2	4.1.2	54	43	13.95	52
20	Samsung	GT-I9505	4.4.2	4.4.2	54	1128	15.37	52
21	LGE	Nexus 4	4.4	4.4	54	5	0.04	52
22	Samsung	SM-G900F	4.4.2	4.4.2	54	205	10.16	52
23	Samsung	GT-S7562	4.0.4	4.0.4	54	7	0.11	54
24	LGE	Nexus 5	4.4.4	4.4.4	54	377	12.15	52
25	Sony	D6503	4.4.2	4.4.4	52	554	10.84	52
26	Samsung	SM-G900F	4.4.2	4.4.2	54	5606	46.84	52
27	HTC	One_M8	4.4.4	4.4.4	54	1206	45.43	52
28	LGE	Nexus 5	4.4.4	4.4.4	54	6	0.28	52
29	Samsung	SGH-I777	4.1.2	4.1.2	54	102	16.24	52
30	LGE	Nexus 5	4.4.4	5	52	2072	32.87	52
31	Samsung	GT-I9300	4.3	4.3	54	0	0.00	54
32	Samsung	GT-I9505	4.4.2	4.4.2	54	2	0.02	52
33	LGE	LG-D855	4.4.2	4.4.2	54	37	2.12	52
34	Samsung	GT-I9300	4.3	4.3	54	2	0.10	52
35	LGE	LG-D855	4.4.2	4.4.2	54	196	8.66	52
36	Samsung	SM-G900F	4.4.2	4.4.2	54	415	10.47	52
37	Sony	D6503	4.4.2	4.4.2	54	66	1.28	52
38	Sony	ST23i	4.0.4	4.0.4	54	541	13.89	54
39	Samsung	Galaxy Nexus	4.3	4.3	54	20	0.49	52
40	LGE	Nexus 5	4.4.4	5	52	905	27.14	52
41	LGE	Nexus 5	4.4.4	5.1	52	5602	398.12	52
42	Sony	D6503	4.4.2	4.4.2	54	258	3.90	52
43	Sony	C6903	4.4.4	4.4.4	54	650	11.28	52
44	LGE	Nexus 5	4.4.4	4.4.4	54	14	0.12	52
45	Sony	C6603	4.4.4	4.4.4	54	238	11.16	52
46	LGE	Nexus 5	4.4.4	4.4.4	54	121	3.95	52
47	Samsung	GT-I9505	4.4.2	4.4.2	54	500	9.96	52
48	Samsung	GT-I9100P	4.1.2	4.1.2	54	4012	143.40	52
49	LGE	Nexus 4	4.4	4.4	54	362	9.29	52
50	LGE	Nexus 4	4.4.4	4.4.4	54	65	1.02	52
51	Samsung	GT-I9506	4.4.2	4.4.2	54	12	0.21	52
52	Samsung	GT-S5830i	2.3.6	2.3.6	54	516	18.07	54
53	Sony	D6503	4.4.4	5.0.2	52	11091	195.29	52

Figure 5.1: Flowchart of the proposed method.



5.4 Experimental Analysis

5.4.1 Data Collection

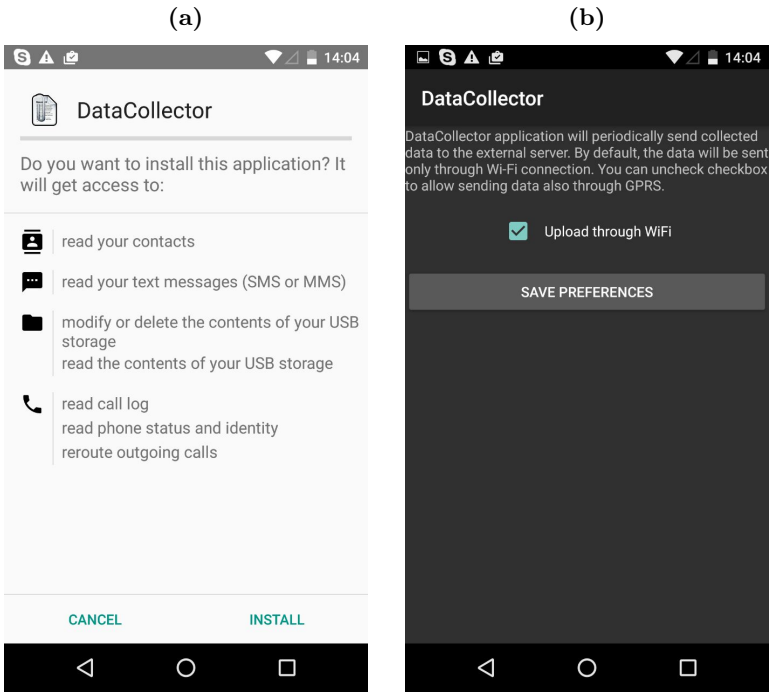
We developed an Android application, namely, DataCollector which collects the data for the analysis. The application is designed to operate in the background (as a separate service), to emulate the behavior of an authentication application.

In order to collect the data for our analysis (supervised learning task), it was necessary to collect user's data during their daily routine of using their smartphone. We set up a webpage which explained the purpose, methodology, and other related details of the experiment and a download link where they could get the DataCollector. Moreover, the DataCollector app itself displayed to users all the above-mentioned details of the experiment. Users could install the application after agreeing a consent form.

DataCollector collects data from multiple sensors, namely, the accelerometer, gravity, gyroscope, magnetometer, and orientation as in Chapters 3 and 4). For each sensory reading, we collect 3-dimensional values denoting the user's motion in a particular dimension, and additionally calculate their magnitude (norm).

Our app gathers information from the sensors with the `SENSOR_DELAY_NORMAL` delay. According to the Android documentation[3], for every sensor, data samples are generated at most every $200,000$ microseconds. Information about system events is recorded as soon as they occur. Every measurement is followed by a timestamp using the sys-

Figure 5.2: Screen shots of our DataCollector app: Figure 5.2a shows the application installer and the Figure 5.2b shows the connectivity manager.



tem call `System.currentTimeMillis()`. Thereafter, collected data are packed into the JavaScript Object Notation (JSON) message and stored as text entries into a file (one file for every sensor). Every two hours our application compresses the collected data to save storage space and sends the encrypted (to ensure data confidentiality) archives to our web server. After each successful transmission, the compressed files on the device are deleted, otherwise, the app keeps retrying.

To ensure participant's privacy, we did not collect any information that can be used to identify a user (e.g., IMEI, IMSI, or phone number). To identify different app instances, DataCollector generates a random unique identifier during the installation. This identifier is later used to label different users on the server. Moreover, our application does not gather any sensitive information, e.g., location, user contacts, etc. To facilitate the user participation, DataCollector was developed with the objective to limit the amount of interactions required to configure the app. User involvement is required only during the installation, initial configuration, and for the uninstallation of the app (see Figure 5.2). Initial configuration only required users to select if data must be transmitted only through WiFi or also using mobile broadband. A total of more than 90GB of raw data were collected.

5.4.2 Feature Extraction

We use statistical features calculated over the sensor measurements gathered within a specified time interval after the `USER_PRESENT` event. We experimented with time interval of 2, 4, 6, 8, 10 seconds. From each sensor data, we extracted 7 statistical values, namely, Mean, Mean absolute deviation (Mad), Median (Med), unbiased Standard Error of the Mean (Sem), Standard Deviation (std), unbiased Skewness (Skew) and kurtosis (Kurt). Thus, for every sensory observation there are 28 extracted features, as listed in Table 5.2).

Table 5.2: List of extracted features from all four dimensions of each sensor.

No.	Features of physical sensors
1	Mean
2	Mean Absolute Deviation (Mad)
3	Median
4	Unbiased Standard Error of Mean (Sem)
5	Standard Deviation (Std)
6	Unbiased skewness (skew)
7	Kurtosis (Kurt)

Normally, the extracted features need to be scaled (or normalized depending on the context) before being processed by machine learning algorithms. However, in our case, we skipped this transformation for two reasons. Firstly, the Android system does not provide an Application Programming Interface (API) to find out the minimum and maximum boundaries of sensor measurements. Hence, the scaling operation will require the authentication application to analyze a large amount of historical data in order to detect the feature values boundaries. This demands additional storage space that is limited in mobile environments. Moreover, it is possible that after the training phase, some outliers may appear in our measurements. Scaled using the learned boundaries, these values will still hugely outperform them, thus, influencing a lot the final decision. Secondly, scaling operations require additional computational resources, which are limited in the case of mobile devices, so our system uses raw feature values.

5.4.3 Feature Subset Selection

To select the best subset, that is the subset which yields maximum accuracy, out of all 196 available features, we relied on InfoGainAttributeEval[2] - a WEKA implementation for Information Gain (IG) based feature selection. This feature selection scheme evaluates the worth of a feature by computing the information gain of that feature with respect to the class. We straight away excluded all the non-contributing features, i.e., having *zero* value (see Figure 5.3). In addition, to avoid any chances of overfitting, we picked 50 top-gain features (marked above the red line), making them equivalent to the number of samples, for further analysis.

[2]http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html.

Figure 5.3: Feature Selection for different time periods, i.e., 2000ms (5.3a), 4000ms (5.3b), and 6000ms (5.3c).

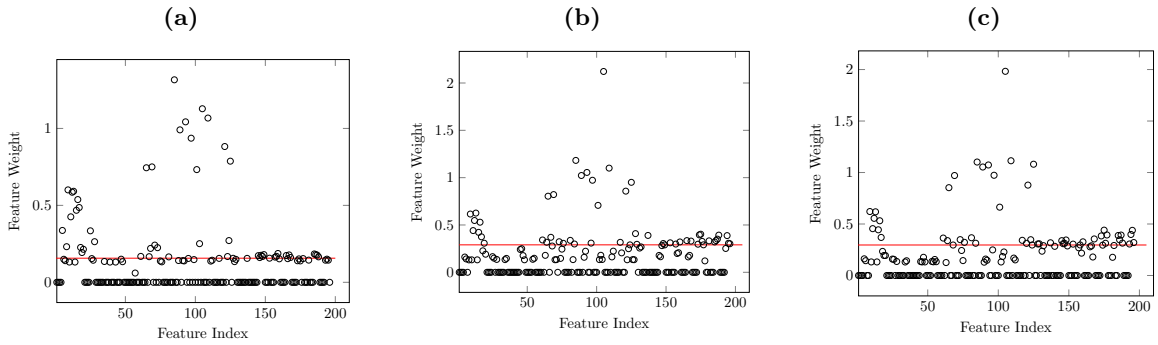


Table 5.3: Results of different classifiers for different lengths of data acquisition (averaged over all 31 users).

	2000ms		**4000ms**		**6000ms**		**8000ms**		**10000ms**	
Classifiers	TAR	EER	TAR	EER	TAR	EER	TAR	EER	TAR	EER
BN	0.89	0.11	0.89	0.11	0.89	0.11	0.88	0.12	0.89	0.12
MLP	**0.93**	**0.07**	**0.93**	**0.07**	**0.94**	**0.06**	**0.94**	**0.06**	**0.94**	**0.06**
1NN	0.88	0.12	0.88	0.12	0.89	0.11	0.89	0.11	0.90	0.10
RF	**0.95**	**0.05**	**0.95**	**0.05**	**0.95**	**0.05**	**0.95**	**0.05**	**0.95**	**0.05**

5.4.4 Validation

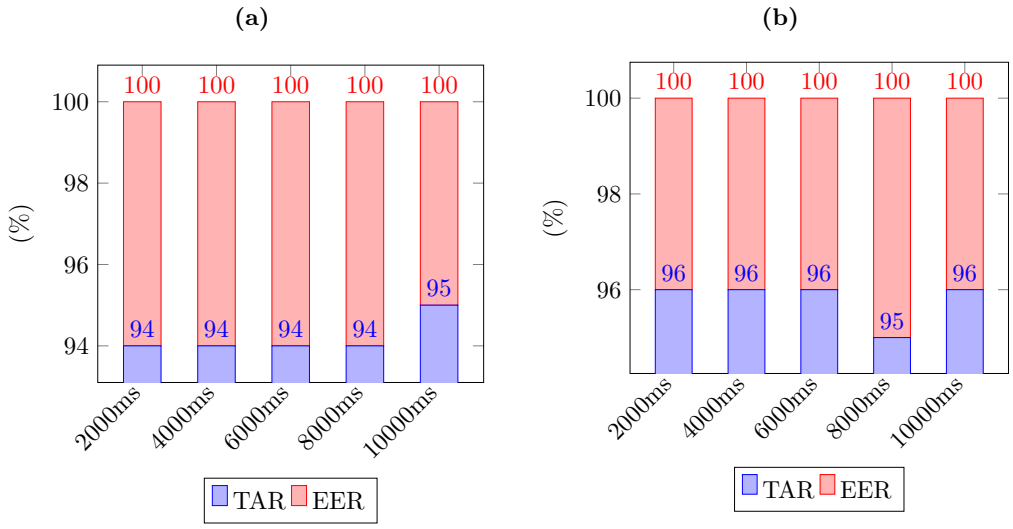
Our experimental validation involves the collection of labeled raw data from multiple 3-dimensional smartphone sensors and then transforming them into the patterns. A pattern here is the horizontal concatenation of all the features of all the sensors (196 before feature subset selection), as discussed in section 5.4.2. The resulting 50 patterns for each of the 31 users are 196 feature long. Note that we take into account only users with ≥ 50 patterns.

Since we have limited number of user patterns (50 only), our analysis is based on 10-fold cross-validation for all experiments with 10 runs. The setting looks justified because in this way, each available sample is tested and their average is reported.

5.5 Results

The results of all of our chosen classifiers before the feature selection are shown in the Table 5.3. We can see that RF and MLP classifier performed best with default parameters (100 trees for RF and 1 hidden layer for MLP) yielding up to 95% and 94%, TAR,

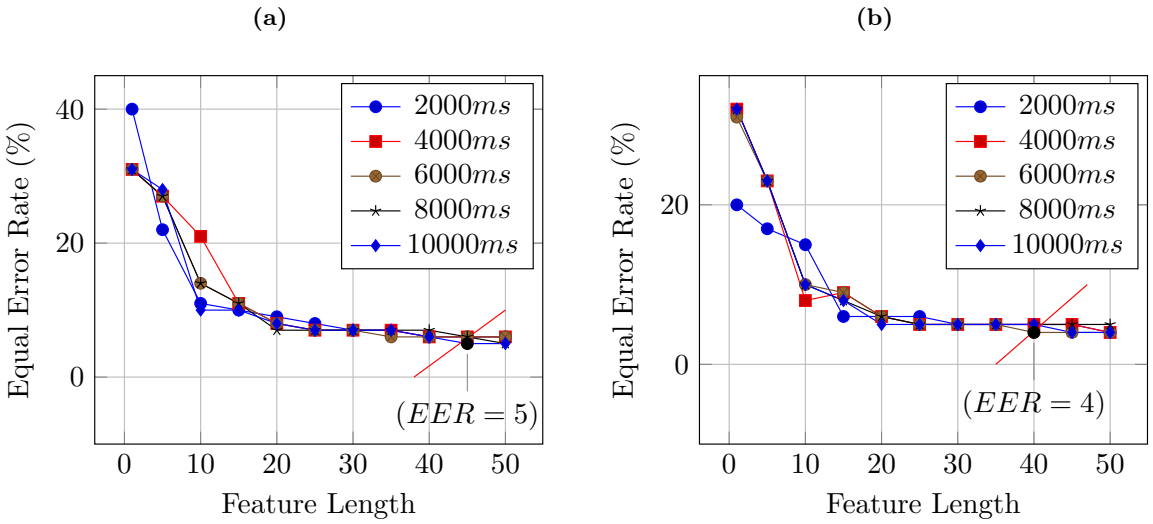
Figure 5.4: Results in terms of TAR and EER (on selected features set) for MLP 5.5a and RF 5.5b classifiers.



respectively. Thus, we take these two classifiers for further analysis. The Figure 5.4 shows the outcome of both MLP and RF classifiers on the subset of selected features. MLP classifier performed best on 10s data yielding 95%, however, RF classifier proved itself consistent on all the available lengths of the dataset.

We further evaluated the different feature lengths in order to (i) cross check our earlier obtained results, and (ii) observe if the same accuracy can be obtained with even less features (see Figure 5.5). The best EER of 5% is obtained for 8s and 10s durations with MLP classifier, however, RF classifier is found consistent with all the durations yielding 4% EER except the 8s time interval. It is also worth-mentioning that for 6s time duration, we obtained 4% EER with just 40 features. Of course, smaller time intervals have to be preferred to long ones, if the accuracy is the same because they are faster and reduce battery consumption. So the best option to choose is the interval of 6s with just 40 features.

Figure 5.5: Results in terms of EER for different feature lengths (from selected features) for MLP 5.5a and RF 5.5b classifiers.



5.6 Discussion

Our participants reported a higher power consumption of about $5-12\%$ measured using the Android's internal power reporter, due to use of the DataCollector app. However, the end system will consume less power because it will collect the sensory readings for smaller time periods, i.e., 6 sec, while DataCollector gathers sensor readings all the time when the screen is on, which is on average equal to 70.3sec [89]. Moreover, we expect in the near future that all mobile platforms will be equipped with low-power continuous sensing modules [90], that will further reduce power consumption. The final implementation and its complete evaluation is a subject of future work.

We assume that during the experiment a smartphone was used solely by the owner. However, in general case this is not true, e.g., sometimes a smartphone may be used by a family member, a friend, etc. We did not apply any outlier detection approach to filter out and delete such outliers. Such filtering should in principle lead to better results.

Our model does not consider the impact of situations while authenticating. As some papers show [29] (see Chapter 3), situations (i.e., walking, standing, running, etc.) may affect the behavioral pattern. If the phone is unlocked while walking the resulting pattern would be different if the same user unlocks the phone while lying on a bed. On a positive side, we tested the system in an uncontrolled fashion so the users were not constrained to a specific situation and data were gathered in a realistic fashion mixing different situations.

Neverthesless it might be interesting to check the impact of each situation on the aggregate results. As a future work, we will extend the DataCollector app to recognize situations (e.g., by using JigSaw engine [91], etc.) and select the most appropriate set of features for that situation.

We plan also to extend the experimental validation with a higher number of testers.

5.7 Chapter Summary

This chapter presented a novel approach for unobtrusive user authentication on smartphone. Our proposed method is based on profiling hand(s) micro-movements, after an unlock event occurs, using smartphone built-in unprivileged sensors. The design allow to implement our method as a separate authentication service, which may be used by different applications (i.e., mobile banking, m-health app, etc.).

We have shown that by profiling the user based on simple time-domain features, extracted from sensory data, we can authenticate the smartphone users with high precision. To validate our approach, we launched an uncontrolled experiment with 31 qualified users (53, in total). We collected real-world readings from all the built-in 3-dimensional sensors (5 physical and 2 LPF and HPF readings) and share this dataset with the research community. Using the obtained data, we inferred critical parameters for our system, e.g., the data collection time interval. We also used the dataset to assess our system. The experiments showed that our proposed approach achieves the TAR of 96% at an EER of 4% in the authentication task.

Chapter 6

ACTIVEAUTH:A Motion-Based One-Shot-cum-Continuous User Authentication Scheme for Smartphones

In this chapter, we present *ACTIVEAUTH* - a fully unobtrusive motion-based one-shot-cum-continuous user authentication scheme for smartphones, which in addition to authenticating the user at login stage (as discussed in Chapter 5), continuously tracks the user interactions and authenticates the user before the user installs an application package or uninstalls an already installed application package. More specifically, *ACTIVEAUTH* starts all the 3-dimensional sensors, records the movements for a short period of time ($5s$), extracts the features and applies a $1-class$ algorithm to verify the identity of the user. *ACTIVEAUTH*, due to its unobtrusive nature, can be used as a standalone solution or can be complemented with any of the existing authentication mechanism to improve the authentication accuracy and robustness.

6.1 Introduction

In this chapter, we present a motion-based fully unobtrusive and hassle-free one-shot-cum-continuous user authentication scheme, namely, *ACTIVEAUTH*, which provides one-shot login as well as can continuously verify the presence of the legitimate user during the entire session, without requiring any user permission, interaction, participation, and co-operation. The proposed approach is based on the idea that every user has unique ways of holding and moving her smartphone [29][5][27][31] (see Figure 6.1) thus generating unique movement patterns. These generated movement patterns can be exploited for

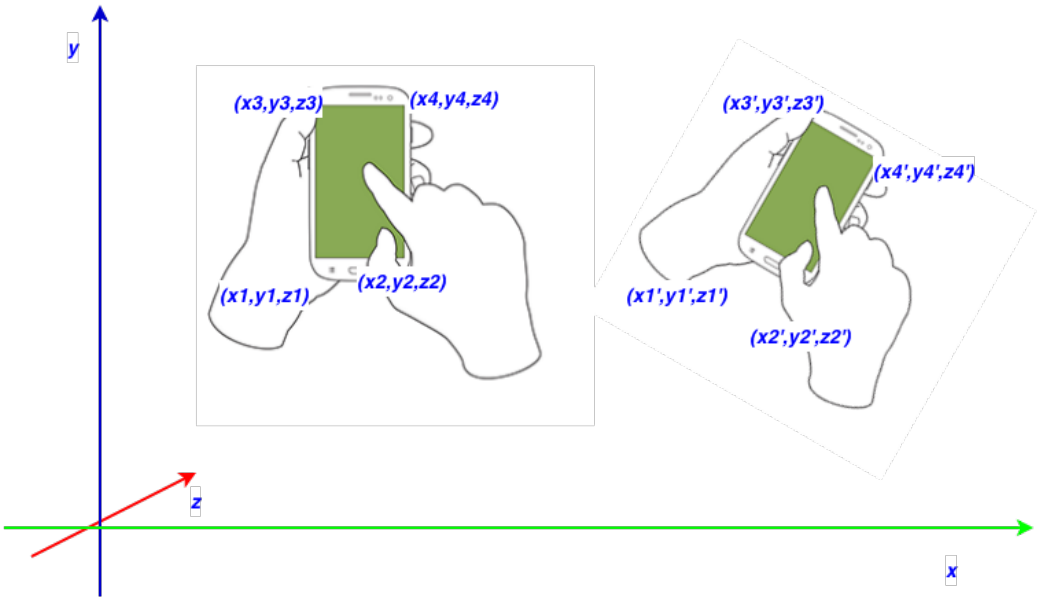


Figure 6.1: Different phone positions for user interaction [5].

user authentication. *ACTIVEAUTH* utilizes 3 unprivileged broadcast receivers, namely, *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* to record the phone-movements and profile the users accordingly. The *USER_PRESENT* broadcast is fired each time the user enters her credentials to unlock their smartphone. Similarly, *PACKAGE_ADDED* and *PACKAGE_REMOVED* notifications are issued each time a new package is installed or an already installed package, is uninstalled. At this stage, we relied on these broadcast receivers because they do not require any user permission and hence the data is collected transparently, while the others broadcast receivers, require user permissions and the user may feel uncomfortable granting the required permissions. We collected sensory data right after the notification of these events, i.e., *USER_PRESENT*, *PACKAGE_ADDED* or *PACKAGE_REMOVED*, for $5s$. We chose a duration of $5s$ for two reasons, i.e., firstly, since the duration could be acceptable to the users (because each authentication method available requires comparatively higher time [5]), secondly, we consider this duration too short for hacking.

ACTIVEAUTH profiles the smartphone user based on the time-domain features (extracted directly from the sensory readings) and builds the authentication model using $1-class$ learning algorithms for each broadcast notification. Based on the collected usable dataset of 80, 50 and 49 qualified users (in total, 123 users) for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED*, respectively, *ACTIVEAUTH* can authenticate the user with 84.06% TAR at an FAR of 19.57% at login stage (with *USER_PRESENT* notification), and for continuous operation, i.e., *PACKAGE_REMOVED* and *PACKAGE_ADDED* broadcast events, it can authenticate the user with a TAR 98%

and 93.88% at an FAR of 14%, and 9.1%, respectively.

6.1.1 Contributions

The main contribution concerning this section are listed below:

- The proposal of *ACTIVEAUTH* - a fully unobtrusive one-shot-cum-continuous user authentication scheme, which can be enabled as an standalone or can be complemented with any of the existing smartphone user authentication approaches for their security and/or usability enhancement.
- The collection and sharing of the collected dataset from 123 users with the research community in order to open new research dimensions.
- Experimental validation on the dataset collected from 123 users.

6.2 Related Work

One-shot authentication schemes have been more studied, tested and evaluated as compared to the continuous schemes. One-shot authentication schemes are designed to authenticate the user at the start of the session - leaving behind the possibility of session hijacking and masquerading. In contrast, continuous authentication schemes are designed to continuously verify the user's identity throughout the session. Most of the proposed authentication solution are based on either face [92] [93], touch [94] or swipe [95] or signature [96], gait [97][98], fusion of multiple modalities [99] [100] [101] [102], and on the device movements [5][31]. However, a little work has exploited only motion sensors for continuous user authentication on smartphones [75][53][81]. Interested readers are referred to this survey [30] for understanding behavioral biometrics, in general, and to [103] for more insights on continuous authentication on smartphones.

In the following sections, we will survey the most relevant sensor-based one-shot and continuous authentication schemes proposed over the years.

6.2.1 Sensors-based One-shot Authentication Schemes

Some of the studies [27] [31] [104] add a transparent layer (using data collected from the smartphone sensors) to the PIN authentication scheme. Our approach [27] (see Chapter 3) extends a classical PIN authentication to a bi-modal system adding the user's hand movement (for the total duration of PIN entry) as a separate modality. we reported an EER of 1%, on a dataset of 12 users, using BayesNET and Random Forest (RF) as classifiers. Authors of this study [104], collected passcode-generated sensory data from motion sensors for the entire duration of passcode entry and profiled the typist. Using

$1-class$ verifiers on the collected data of 48 users, they report a TAR of 93.15% and an FAR of 5.01%. Similarly, our other former study [5] (see Chapter 4) proposes a bi-modal system based on the smartphone's micro-movement when the user writes something on the touchscreen. In this way, we train our classifier (Multilayer Perceptron) on touch and motion events, generated while the user writes something on the touchscreen. We reported a TAR of 94.8% and an FAR of 3.1%.

6.2.2 Sensor-based Continuous Authentication Schemes

Lee et al. [75] presented a continuous and transparent motion-based authentication solution for user authentication on smartphones. Their approach leverages three smartphone sensors, namely, the accelerometer, the magnetometer and the orientation, however, they drop the orientation sensor, later, because they observed it as less accurate and hence less productive for their multi-sensor system. Their mechanism implicitly profiles the user based on her movements and by using SVM as a binary class classifier, and later authenticate the user. Their mechanism requires $10s$ to train the model for a smartphone user and needs $20s$ to detect the attacker. They report an overall accuracy of 90%. The downside of this study is their limited number of users, i.e., just 8.

In a recent study, Shen et. al. [107], proposed a sensor-based continuous authentication framework and analyzed its performance on sensor-assisted touch-tapping data collected from 50 users. Authors utilize three physical sensors, i.e., gravity, accelerometer and gyroscope to profile user's movements. They applied $1-class$ SVM and KNN as verifiers, and achieve an EER of 11.05%.

Our proposed approach, namely *ACTIVEAUTH* - a completely unobtrusive, friction-less, light-weight, and user-friendly one-shot-cum-continuous authentication scheme, does not require any user permission, participation or cooperation. *ACTIVEAUTH* is context-based - it collects and identifies user whenever needed, i.e., when a user installs a package or uninstall any installed package from the system. The authentication cycle starts with the user unlocking their phone (one-shot) and keeps tracking the installation and removal of packages till the session expires. *ACTIVEAUTH* performs one-shot authentication when the user unlocks the smartphone by either applying any existing authentication mechanism or slide-to-unlock her smartphone. In any case, *USER_PRESENT* broadcast is issued. *ACTIVEAUTH* then turns on all the 3-dimensional sensors for a short period of time (i.e., $5s$), extracts time-domain features from the recorded sensory reading for that time period, and authenticates the user based on the similarity of the sample with the training samples. Access to the smartphone is granted once the user is authenticated. In this way, *ACTIVEAUTH* ensures the presence of the genuine user at one-shot login. Then *ACTIVEAUTH* keeps tracking the user interactions and the process of authentication is

Table 6.1: Comparison of our authentication mechanism with the related work. Our comparison is limited to the work which involves sensory readings and user interaction with the device, i.e., tapping, touch, etc. Ac, Os, Gr, Gy, Mag stand for accelerometer, orientation, gravity, gyroscope and magnetometer, respectively. Similarly, BN, RF, SM, SE, SVM, KNN, MLP stand for bayesNET, Random Forest, scaled manhattan, scaled euclidean, support machine classifiers, K-nearest neighbor and Multilayer erceptron. O denotes the smartphone owner and I denotes Impostors. O & I mean the system was trained with data from owner and impostors.

Paper	Form	Input Method	Sensors	Classifiers	Users	Data Source	Results
[54]	Continuous	touch + movement	Ac, Os, compass and touchscreen	SVM	28	O & I	accuracy = 95.78%
[81]	Continuous	touch + movement	Ac, MIC,Location and touchscreen	NaiveBayes	7	O & I	accuacy = 97%
[53]	Continuous	movement	Ac, Os, Gy and Mg	n-gram	20	O & I	accuracy = 71.3%
[105]	Continuous	touch + movement	Ac,Rotation	SVM	100	O, O & I	0% - 24.99% FAR
[106]	Continuous	touch + movement	Ac,Rotation	SVM	300	O & I	TAR = 92% - 1% FAR
[31]	Continuous	tap + movement	Ac,Gy,Mag	SM,SE,SVM	100	O	EER = 6.92%
[28]	One-shot	movement	Ac, Os	DTW	10	-	TAR = 90.77%, FAR = 4.44%
[27]	One-shot	tap + movement	Ac (3 variants),Os,Gr,Mag	BN & RF	12	O & I	EER = 1%
[29]	One-shot	movement + voice	Ac, Os, Gr	BN,SVM,RF	26	O	TAR = 95.8% , FAR 11.01%
[104]	One-shot	touch + movement	Ac, Gy	BN,SVM,RF	48	O	TAR = 93.15% , FAR 5.01%
[5]	One-Shot	touch + movement	Ac (3 variants),Os,Gr,Gy,Mag	MLP	30	O	TAR = 94.8%,FAR = 3.1%
This work	**One-shot Continuous**	**movement**	**Ac,Gy,Mag,gr, Os**	**MLP, FRF, Gauss_DD**	**123**	**O**	**TAR = 84.06%, 98%, 93.88%**

repeated if the users installs (as a result of *PACKAGE_ADDED* notification) any new package or uninstalls (as a results of *PACKAGE_REMOVED)* any installed package. It is worth mentioning that none of these broadcast receivers require any Android permission. We could have used multiple broadcast receivers for continuous authentication, however, at the cost of user permission and more computations. Our idea is to ensure the user authenticity unobtrusively - without the need for user participation even for the grant of permission. Another notable difference is the nature of *ACTIVEAUTH* - Most of the proposed schemes are either one-shot or continuous, however, *ACTIVEAUTH* covers both functionalities simultaneously. Another difference between our approach and the state-of-the-art is the classification, i.e., we deal this as $1-class$ user verification problem (owner Vs attackers) whereas most of the above-mentioned approaches cater this as the binary-class classification problem, which seems unrealistic because training the classifier with patterns of two users may lead to privacy and security concerns[31][5].

Since the concept behind *ACTIVEAUTH* is novel, and we may not directly compare our work with the previous studies, however, we present an overview of some of the most relevant motion-based state-of-the-art one-shot and continuous authentication approaches and highlight the key aspects of each considered work in the Table 6.1. We compare them in terms of: (i) the nature of the scheme, i.e., one-shot or continuous, (ii) the input method (was it only motion-based or was complimented with another input method), (iii) considered sensors, (iv) classifiers, (v) no. of users. (vi) classification approach, and (vii) obtained results.

6.3 Motion-based One-shot-cum-Continuous Authentication Scheme

We present a motion-based one-shot-cum-continuous smartphone user authentication scheme, *ACTIVEAUTH*, which starts verifying the user's identity right after receiving an Android OS notification of *USER_PRESENT* broadcast receiver. In this manner, the proposed approach ensures one-shot authentication. It is notable that this broadcast receiver is fired each time the user either enters her credentials, e.g., passcodes, or performs slide-to-unlock gesture for login. The proposed approach is equally useful for the smartphones with or without any implemented authentication mechanism. An attacker has to pass this authentication mechanism too, besides another authentication requirement, e.g., passcodes. This adds a transparent authentication layer to the existing authentication approach.

Besides providing a one-shot login, our proposed approach later monitors the user activities, i.e., addition or removal of a package. We profile user's hand-movements when she starts the package installation and uninstallation. In either case, it transparently profiles the user using motion data collected from all the physical sensors and authenticates the genuine user. In case it detects the genuine user, it grants the access to the system and consequently lets the user to install/uninstall, otherwise, it asks for the explicit authenti-

cation, e.g., passcodes, etc. We could have used multiple broadcast receivers (associated with different user activities, e.g., sending SMS, making a call, etc.), however, due to their requirement for user permission, we avoided them. The same can be developed for security-conscious users.

This section explains various building blocks of *ACTIVEAUTH* - the proposed one-shot cum continuous authentication scheme as depicted in the Figure 6.2.

6.3.1 Broadcast Receivers

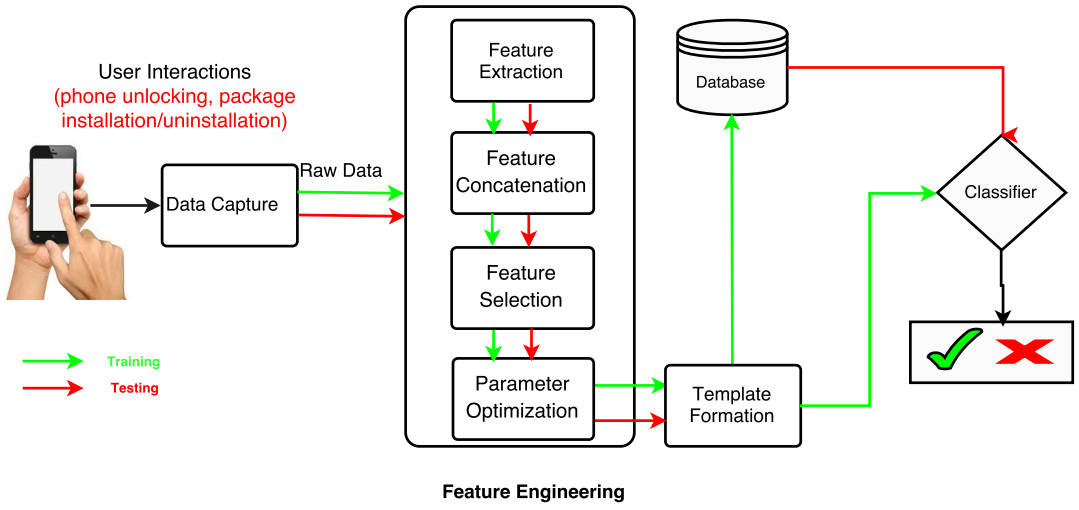
In Android, the broadcast receiver allows the developer to register for the system and application events, i.e., Operating System (OS) will keep looking for the registered event(s) and notify time to time whenever the subject event(s) occur. For example, the reception of SMS, change of battery status, availability of Wi-Fi, screen ON/OFF, etc. All these events are system generated events.

Similarly, the developed application can also initiate such broadcasts. All chosen broadcast events require the presence of the user and their input to perform the desired task. In the following, we explain the preliminary selected broadcast receivers enabled in *ACTIVEAUTH*:

- *USER_PRESENT*: Every smartphone user interaction session starts after the screen in turned ON. Screen can be turned ON either (i) by pressing the power button, or (ii) by the phone ringing. Turning the screen ON by power button brings the unlock screen up front, and requires the user to enter either her registered credentials or perform the slide-to-unlock action to start the session. In either case *USER_PRESENT* notification is fired. This broadcast ensures the user presence and their interaction with the device. We consider this as the best time to profile the user based on her movements after *USER_PRESENT* event is fired. The proposed method collects the sensory data for a short duration, i.e., $5s$ for fingerprinting the user's movement and authenticate them using 1-class verifier, and in this way adds a transparent layer to the implemented authentication mechanism.

- *PACKAGE_ADDED*: This broadcast notifies the successful installation of an application package on the device. We are of the opinion that an attacker can install some application to quickly retrieve all the user's sensitive private information, e.g., login credentials for banking, and social sites, etc. for misuse. At this point, our proposed method ensures whether or not the user is legitimate. Within a very short period after this notification, it verifies the user identity and permits the user to open this newly installed application, otherwise, the application is blocked (cannot be opened) and an explicit login is required to open it.

Figure 6.2: Block diagram of our proposed approach.



- *PACKAGE_REMOVED*: This broadcast notifies the completion of uninstallation of an application package. We consider the application uninstallation as a threat. It is possible that the smartphone has an anti-virus or anti-theft[1] application package already installed, which attacker may want to uninstall. To secure that application package, *ACTIVEAUTH* first creates the backup of that application and then allows the uninstallation. Having the app temporarily uninstalled, the proposed approach authenticates the user, i.e., it can be removed permanently in case the user is found legitimate. In case of an impostor, it rolls back the uninstallation in background while asking for explicit user login in the foreground.

The reasons behind the selection of the above-mentioned broadcast receivers are: (i) they actually require the user presence and interaction, and, (ii) they do not require the user permission (hence avoiding their cooperation).

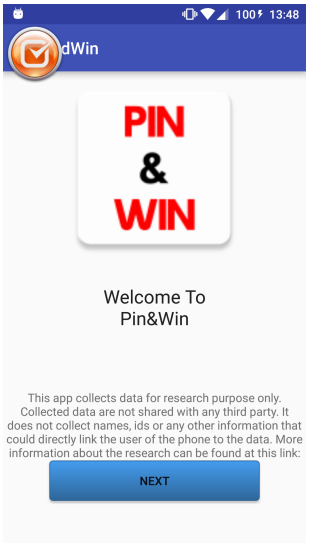
6.3.2 Data Collection

We prototyped an Android application *PIN&WIN* to collect data for our analysis. Our application can be installed on any Android device running 4.4.x OS or higher (see Figure 6.3). The application was developed mainly for the analysis of sensor-assisted touchstrokes [27] (see Chapter 3) for smartphone user authentication. We embedded these broadcast receivers into our *PIN&WIN* application with the objective to transparently collect user's

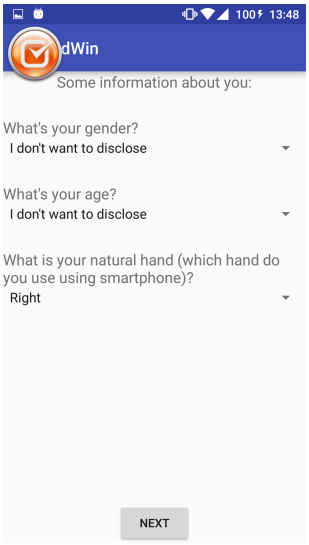
[1] http://getandroidstuff.a/cerberus-best-android-anti-theft-app/

Figure 6.3: The screenshots of *PIN&WIN*.

(a) Intro & consent agreement



(b) Demographic Questionaire



data. In order to obtain a realistic view, e.g., how often they install and/or uninstall the application package(s), how do they install and/or uninstall a package, etc., we kept this functionality hidden from the volunteers. We setup a web-page[2] with the complete explanation of *PIN&WIN*, i.e., the user consent, the procedure to install/uninstall the application and the incentive. Participants were requested to install the application, answer to the demographic questions (see appendix B), enter $8 - digit$ PIN/password in different activities and keep the application running for at least 3 days. PIN&WIN requires user's interaction in 3 sessions in 3 days. *PIN&WIN* requires 30-minutes of user interaction on the first day (after installation), and 15 minutes of interaction on the following two days. In this manner, each user had to test the application for 1 hour (in 3 days) and we expected some addition and/or removal of packages in this duration along with the user presence. In this way, the application transparently collects the data related with each of the broadcast receivers. Data was collected in a totally uncontrolled manner.

Our demographics questionnaire comprised of 4 questions (as shown in appendix B) and the corresponding demographics is tabulated in the Table 6.2. Volunteers were free to answer those or choose if they did want to disclose.

A small incentive, i.e., a lunch voucher, was offered as the compensation at the end of the experiment. We advertised our experiment on the public places of our university,

[2]http://titan.disi.unitn.it/experiment/index.html

Table 6.2: User demographics (M = Male, F = Female, U = Undisclosed, R = Right, L = Left, B = Both).

No. of Subjects	$93M$	$24F$	$6U$	-	-
Hand Preferences	$112R$	$6L$	$1B$	$2U$	-
Age Groups	$5(\leq 20)$	$65(20-40)$	$5(41-60)$	$0(\geq 61)$	$5U$

e.g., notice boards, elevators, main doors, etc.

In total, 123 users downloaded the application and installed it on their smartphones. However, we could utilize data only from 80, 50, and 49 qualified users for the chosen broadcast receivers, i.e., *USER_PRESENT*, *PACKAGE_REMOVED*, and *PACKAGE_ADDED*, respectively. Data from other users were discarded for various reasons: (i) some of the users did not complete the study, (ii) some of the smartphones did not have all the required sensors, (iii) some of the users did not install or uninstall any package during the experiment, and (iv) some of the user had task killer and app killers installed on their smartphones (for saving battery drain), etc. So we included the users who have ≥ 40 samples for *USER_PRESENT* and ≥ 10 for *PACKAGE_REMOVED* and *PACKAGE_ADDED* broadcast events, each.

6.3.3 Motion-based Sensory Features

6.3.3.1 Feature Extraction

Our proposed approach utilizes all the available smartphone built-in sensors, i.e., the accelerometer, the gravity, the gyroscope, the magnetometer and the orientation sensor, to profile the user movements. Data collection is started for $5s$ whenever any of the mentioned events is fired. Since, all the sensors are 3-dimensional, the recorded sequences are stored as a tuple, i.e. as X, Y and Z dimensions.

These stored data sequences can be compared directly using the time series analysis approach, e.g., DTW [28], etc. Alternately, the most common scheme is the extraction of features from these sequences. We extracted the descriptive features, namely, mean, standard deviation, skewness, and kurtosis from each of the sequence for each sensor. In addition, we computed the fourth dimension for all of these and call it as *magnitude* as in our study[27] (see Chapter 3).

To summarize, we extracted 16 features per sensor ($16x7 = 112$, in total), for each movement behavior/observation. The selected features are tabulated in the Table 6.3. We concatenated all the extracted features from all the sensors to form a final feature vector.

Table 6.3: List of selected features from 3-dimensional sensors data. The X in the format X_Mean denotes name of the sensor, e.g., Accelerometer, LPF, HPF, and so on.

Feature position	*X*	*Y*	*Z*	*S_M*
1 – 4	X_Mean	X_Mean	X_Mean	X_Mean
5 – 8	X_Std	X_Std	X_Std	X_Std
9 – 12	X_Skew	X_Skew	X_Skew	X_Skew
13 – 16	X_Kurt	X_Kurt	X_Kurt	X_Kurt

Table 6.4: List of CSE selected features for all the broadcast receivers.

Sr. No.	Features (USER_PRESENT)	Features (PACKAGE_REMOVED)	Features (PACKAGE_ADDED)
1	Accel_Mean_S_M	Accel_Mean_S_M	Accel_Mean_S_M
2	Accel_Z_Mean	Accel_Y_Mean	LPF_Mean_S_M
3	LPF_Mean_S_M	LPF_Mean_S_M	Magnet_Mean_S_M
4	Magnet_Mean_S_M	Magnet_Mean_S_M	Magnet_Z_Mean
5	Magnet_Y_Mean	Magnet_Y_Mean	Magnet_Y_STD
6	Magnet_Z_Mean	Magnet_Z_Mean	Gyro_Mean_S_M
7	Magnet_STD_S_M	Magnet_STD_S_M	Gravity_Mean_S_M
8	Gravity_Kurt_S_M	Gravity_Kurt_S_M	Orientation_Kurt_S_M
9	Gravity_Mean_S_M	Gravity_Z_Mean	Orientation_Y_Mean
10	Gravity_STD_S_M	Gravity_STD_S_M	Orientation_Z_Skew
11	Orientation_Kurt_S_M	Orientation_Kurt_S_M	Orientation_X_STD
12	Orientation_Skew_S_M	-	Orientation_Z_STD
13	Orientation_Z_Skew	-	-

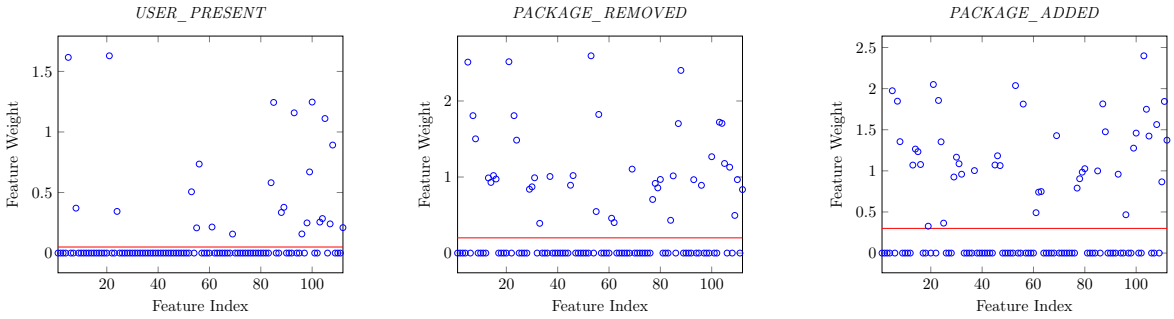
6.3.3.2 Feature Selection

Our motivation to incorporate the feature selection strategy is to reduce the algorithm training time and computational cost because the proposed approach targets the smartphones - as they have limited resources compared to laptops/desktops. We tried 2 WEKA-implemented feature selection schemes, namely, CfsSubsetEval(CSE)[3] and InfoGainAttributeEval(IGAE)[4].

CSE feature selection scheme ranks different features based on their predictive power along-with the degree of redundancy between them [108]. We applied CSE scheme with `BestFit` bidirectional search method to find the best subset. `BestFit` searching algorithm searches the whole attribute space and find the best (the most meritorious) feature subset by applying greedy hillclimbing approach. The scheme evaluated, in total, 1792 , 2128, 2352 subsets and found a best subset of 13, 11 and 12 features for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED*, broadcast receivers (see the Table

[3] http://www.dbs.ifi.lmu.de/ zimek/diplomathesis/implementations/EHNDs/doc/weka/attributeSelection/CfsSubsetEval.html
[4] http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html

Figure 6.4: Feature Selection from *USER_PRESENT*, *PACKAGE_REMOVED* *PACKAGE_ADDED* broadcast event data using IGAE.



6.4).

IGAE - a mutual-information based feature selection scheme, ranks the features based on their contained information gain with respect to the class. The outcome of this scheme provides the feature ranking based on their feature weights (higher the better and *zero* is non-productive). We filtered out all the non-productive features and formed a feature subset with the *non-zero* feature-weight features- features above the read line are used for further analysis for the 3 broadcast events, i.e., *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* (see Figure 6.4). More specifically, our selected feature subset becomes 39, 51 and 53 features long for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* events, respectively.

We used all of our data as training set for performing feature selection. The reason behind this setting is the fact that we used classifier-independent feature selection schemes (which do not involve any classifier) avoiding any possible overfitting.

6.3.4 User Authentication Model

Smartphone user authentication has essentially been a binary class classification problem (owner Vs impostor). However, due to the availability of only owner data, the 1-class approach is considered as more appropriate (as model training with owner and impostor may lead to the privacy concerns). Therefore, the realistic approach is the training of the authentication model only on the legitimate user's data and apply a novelty detection approach (1-class classification) to detect the impostors. As such, we deal the smartphone user authentication as a 1-class classification problem [31][5][29].

Our authentication model, for each query attempt, detects whether it is a normal one (from the owner) or an anomalous one (from the impostor) by comparing it with each of the probe samples in the database. The decision is made on the basis of the difference between the query and probe samples, i.e., less different samples are accepted as coming

from the legitimate users.

6.3.4.1 Model Training/Testing

The process of model building starts by profiling one of our users as the owner, and the remaining users as impostors (as in the Chapter 4). We train and test each of our considered verifiers in two scenarios, i.e., owner verification scenario, and impostors detection scenario. In owner verification scenario, the system is trained on the training data of the "owner" class and then tested with the testing data of the same class (owner). This setting provides the binary outcome, i.e., accept or reject. Due to the limited number of observations per user, we consider cross-validation (with $K = 10$) as the appropriate approach for model training and testing. A cross-validation method randomizes the data and splits it into 10 equal folds. In each iteration, one of the fold is used for testing, and the remaining folds are used for training the classifier. This approach looks justified because in this way each observation gets tested. At the end, all the results are averaged over all the folds and the cross-validation accuracy is reported. We report all accepted results under TAR and all rejected results under FRR.

Similarly, in the impostor detection scenario, the model is trained on the observations of owner data and tested with the data of the remaining users (impostors). This setting also results in two outcomes, i.e., FAR and TRR - all accepted attempts are marked under FAR and all rejected attempts are marked under TRR. This testing is repeated for all the users and average results are reported.

6.3.5 Verifier Selection

The choice of the verifier is always dependent on the nature of the problem. Since the work targets smartphones, we focused on the recently proposed verifiers [5][27], i.e., $1-class$ Multilayer Perceptron(MLP), $1-class$ Fast Random Forest (FRF), and $1-class$ Gaussian Data Description ("Gauss_DD[5]") verifier.

MLP is part of the Artificial Neural Network (ANN) family and FRF belongs to the decision tree family. RF[6] verifier grows multiple classification trees and each input feature vector is fed to these trees and asked to predict the label for input feature vector on the basis of majority voting. More classification trees may increase the accuracy, however, at the cost of more memory. Another motivation to include this verifier is its ability to deal with a large number of features.

Gauss_DD models the target class as a Gaussian Distribution and instead of density estimation, it applies the Mahalanobis distance [109] (as per the equation 6.1) to classify

[5]http://homepage.tudelft.nl/n9d04/functions/gauss_dd.html

[6]https://www.stat.berkeley.edu/ breiman/RandomForests/cc_home.htm

the incoming sample (as per the equation 6.2).

$$f(x) = (x - \mu)^T \Sigma^{-1} (x - \mu) \tag{6.1}$$

$$f(x) = \begin{cases} target & \text{if} \quad f(x) \leq \theta \\ outlier & \text{if} \quad f(x) > \theta \end{cases} \tag{6.2}$$

where μ, Σ and θ are the mean, covariance matrix and the threshold for decision making, respectively.

6.3.6 Verifier's Parameter Optimization

This section outlines the steps taken in order to identify the best parameters for all the verifiers. As CSE features have performed well, we will limit the verifier optimization on that feature subset only. We use the same protocol (i.e., owner Vs all would-be attackers) for evaluating different parameters for all the verifiers.

For designing an MLP neural network, the question *How many hidden layers?* is very important and the solution depend mainly on the characteristics of the dataset, e.g., is the data linearly separable? For linearly separable dataset, the default settings, i.e., one hidden layer, could provide the optimum results. However, because of the presence of non-linear data, we tried a number of hidden layers between 1 : 10 to find the best parameter.

The study carried out by Oshiro et al. [110] showed the importance of the number of trees in a forest, i.e., they reported that a large number of trees in a forest does not always provide significant performance improvement, while it does increase the computation cost, i.e., memory, processing, etc. We tested 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024 number of trees.

The implementation of Gauss_DD verifier in DD_Tools [109] does not have any specific optimization parameter rather a regularization parameter, which is used to add some regularization to the estimated covariance matrix to increase the accuracy. The default regularization parameter is set to 0.001. We tried different parameters, i.e., 0.0000625, 0.000125, 0.00025, 0.0005, 0.001, 0.002, 0.004, 0.008, 0.016, 0.032, for fine tuning. Our obtained results for this verifier are illustrated in the Figure (6.10, 6.11, and 6.12). X-axes values are replaced with their logarithmic equivalent for better readability.

Table 6.5: Authentication results (in %) averaged over all 80, 50 and 49 users for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* broadcast events, respectively.

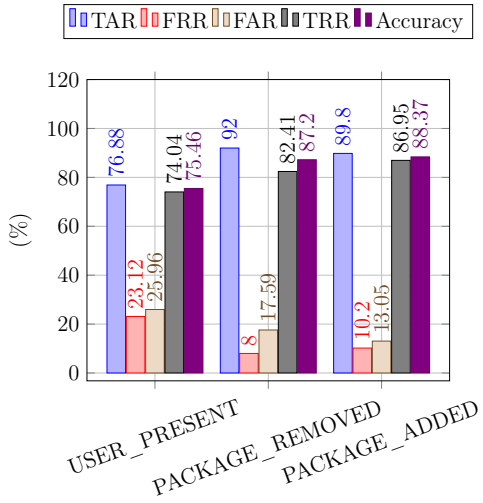
	USER_PRESENT			PACKAGE_REMOVED			PACKAGE_ADDED		
Classifiers	TAR	FAR	Accuracy	TAR	FAR	Accuracy	TAR	FAR	Accuracy
MLP	80.37	34.84	72.76	42.20	9.69	66.25	41.83	6.23	67.80
FRF	79.75	34.62	72.56	41.80	9.60	66.10	41.63	6.00	67.82
GAUSS_DD	**66.56**	**42.29**	**65.50**	**70.00**	**31.67**	**69.17**	**69.39**	**32.31**	**68.50**

6.4 Results

We report our obtained results for each of the chosen broadcast receivers, i.e., *USER_PRESENT*, *PACKAGE_REMOVED*, and *PACKAGE_ADDED*. We report accuracy along-with the TAR and FAR values in order to avoid the redundancy, i.e., as $FRR = 1 - TAR$, $FAR = 1 - TRR$.

We illustrate the obtained results for three broadcast receivers in the Table 6.5 (before applying any feature selection/parameter optimization technique). The Figures (6.7 - 6.9) show the comparison of performance of MLP and FRF verifiers, and (6.5 & 6.6) for Gauss_DD verifier over the chosen IGAE and CSE features.

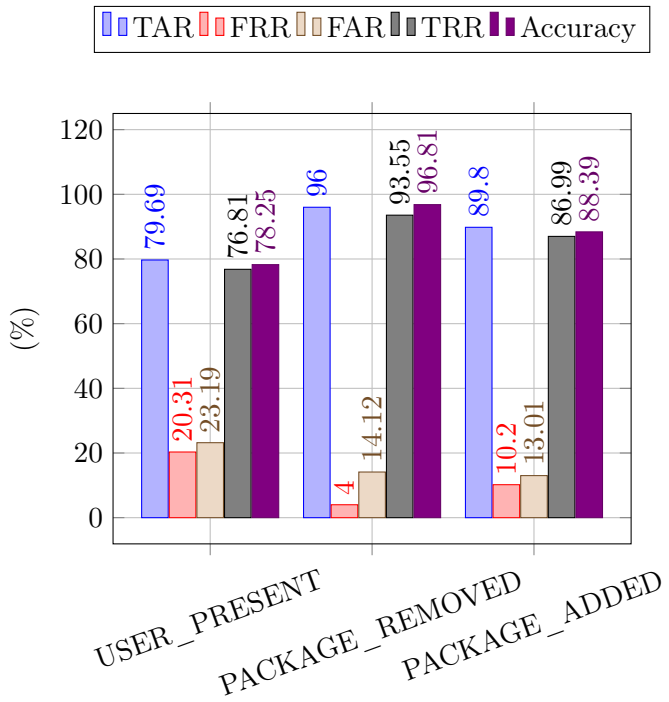
Figure 6.5: Results of GAUSS_DD verifier (with default regularization parameter) with IGAE features. Results are averaged over 80, 50 & 49 users, respectively.



In the tables (see 6.6 & 6.7), we show the obtained results for MLP and FRF verifier for different optimization parameters. Similarly, we show the results of a Gauss_DD verifier on different regularization parameters in the Figures 6.10, 6.11, and 6.12. We have focused more on the best performing verifier, i.e., Gauss_DD, for showing up the results. For example, for MLP and FRF verifiers, we show only the achieved TAR (see figures 6.7, 6.8, and 6.9) whereas for Gauss_DD verifier, we show the results in terms of TAR, FRR, FAR, TRR and accuracy in order to show a clear picture.

We observed a gradual increase in the verifier's performance from the full features to feature selection and parameter optimization. With the full feature set, MLP performed comparatively better yielding 80.37% for *USER_PRESENT* followed by FRF

Figure 6.6: Results of GAUSS_DD verifier (with default regularization parameter) with CSE features. Results are averaged over 80, 50 & 49 users, respectively.



with a TAR of 79.75%. However, both the classifiers were not found useful for *PACKAGE_REMOVED* and *PACKAGE_ADDED* receivers because of less number of available samples (just 10 for each). Gauss_DD verifier remained consistent for full feature sets of *USER_PRESENT*, *PACKAGE_REMOVED*, and *PACKAGE_ADDED*, respectively, which shows its robustness against the less number of training samples. Additionally, it outperformed both MLP and FRF verifiers on the selected IGAE and CSE features as well (see Figures 6.7, 6.8, 6.9, 6.5 and 6.6). The most accurate verifier, i.e., Gauss_DD with default parameters, using CSE (the most productive feature subset), yielded a TAR of 79.69%, 96%, and 89.9%, at an FAR of 23.19%, 14.12%, and 13.01% for *USER_PRESENT*, *PACKAGE_REMOVED*, and *PACKAGE_ADDED* broadcast receivers, respectively. While this TAR further improves to 84.06%, 98%, and 93.88% with a decrease in FAR (from 23.19% to 19.57%) for *USER_PRESENT*, (14.12% to 14.00%) for *PACKAGE_REMOVED*, and (13.01% to 9.1%) for *PACKAGE_ADDED* with the best chosen regularization parameter.

Figure 6.7: Comparison of the obtained TAR for Full, IGAE, and CSE based feature subsets for MLP and FRF verifiers for *USER_PRESENT* dataset.

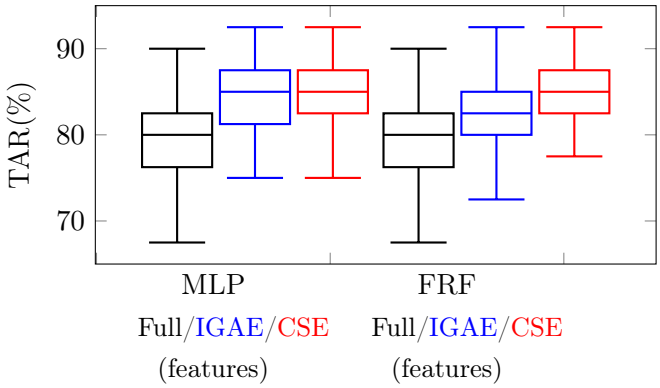


Figure 6.8: Comparison of the obtained TAR for Full, IGAE, and CSE based feature subsets for *PACKAGE_REMOVED* dataset.

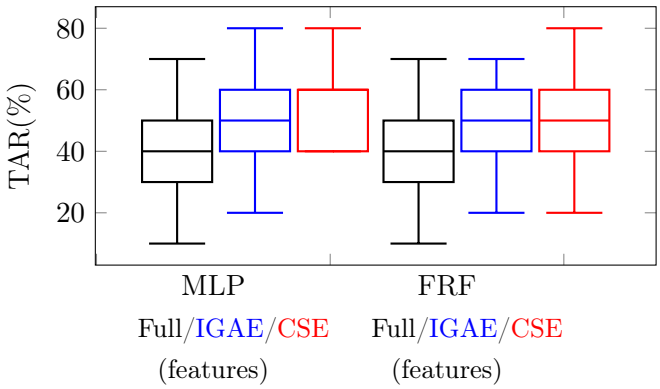


Figure 6.9: Comparison of the obtained TAR for Full, IGAE, and CSE based feature subsets for *PACKAGE_ADDED* dataset.

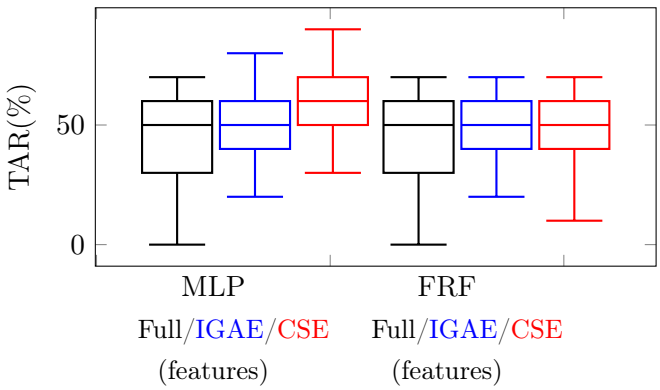


Figure 6.10: Parameter Optimization: GAUSS_DD performed well with 0.0000625 regularization parameter for *USER_PRESENT* broadcast receiver (TAR = 84.06%, FAR = 19.57% and accuracy = 82.24%). Obtained results are averaged over 80 users.

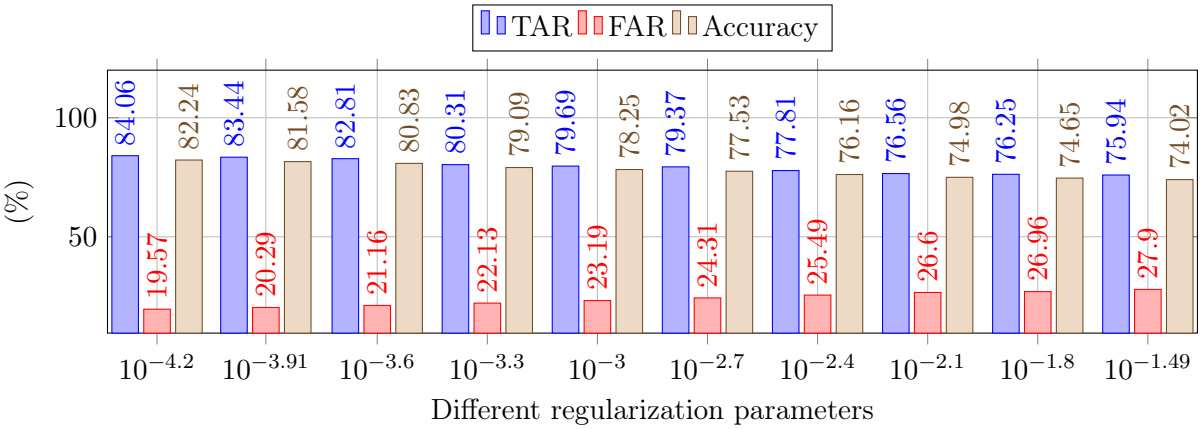


Figure 6.11: Parameter Optimization: GAUSS_DD performed well with 0.004 regularization parameter for *PACKAGE_REMOVED* broadcast receiver (TAR = 98%, FAR = 14% and accuracy = 92%). Obtained results are averaged over 50 users.

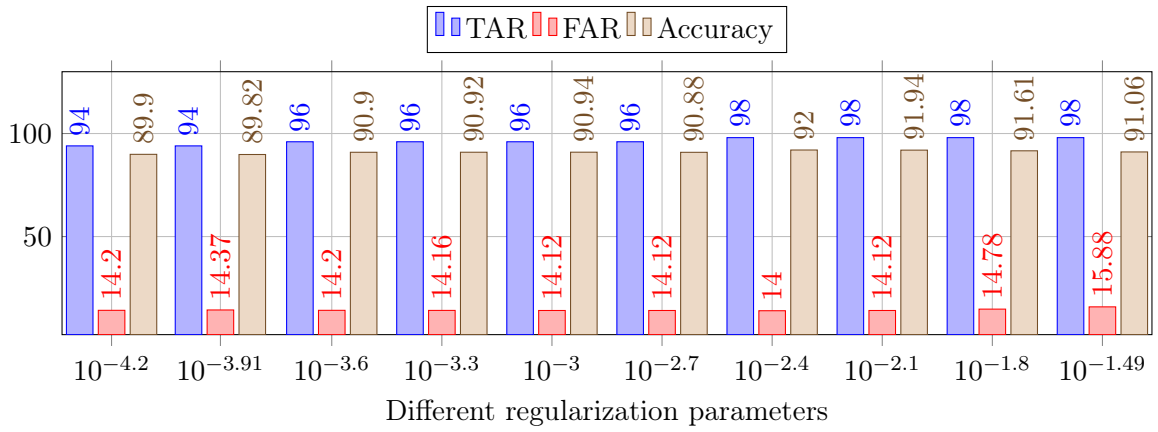


Figure 6.12: Parameter Optimization: GAUSS_DD performed well with 0.0000625 regularization parameter for *PACKAGE_ADDED* broadcast receiver (TAR = 93.88%, FAR = 9.1% and accuracy = 92.36%). Obtained results are averaged over 49 users.

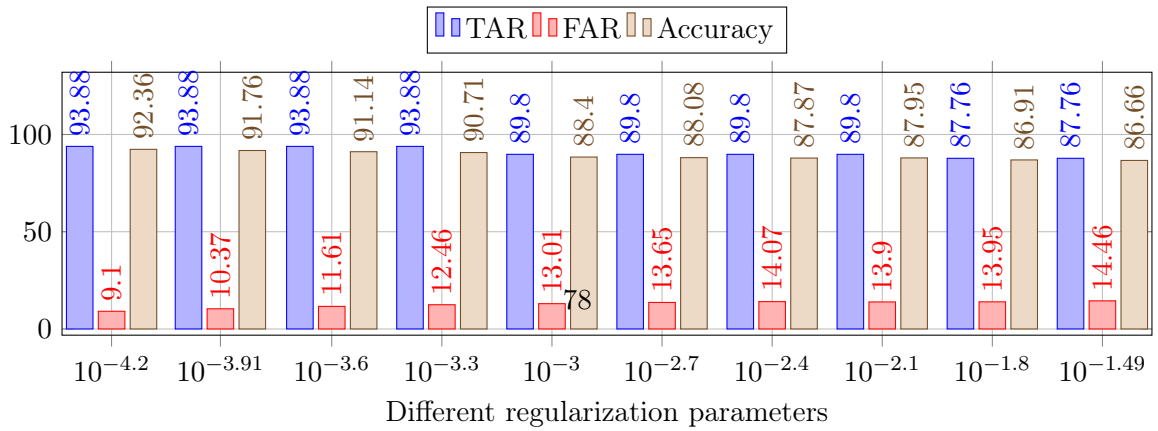


Table 6.6: Authentication results of MLP verifier (in %) for different number of hidden layers averaged over all 80, 50 and 49 users for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* broadcast events, respectively.

	USER_PRESENT		*PACKAGE_REMOVED*		*PACKAGE_ADDED*	
#layers	TAR	FAR	TAR	FAR	TAR	FAR
1	84.97	25.51	55.40	5.71	56.12	3.81
2	86.25	25.51	58.40	5.70	61.22	3.86
3	85.78	26.23	59.80	5.62	61.42	4.11
4	86.03	26.04	59.80	5.77	61.22	4.47
5	86.84	25.93	59.60	5.77	62.44	4.40
6	86.72	25.80	60.00	5.74	62.65	4.59
7	86.69	25.67	60.80	5.96	62.04	4.59
8	86.63	25.70	60.40	5.96	62.86	4.72
9	**86.84**	**25.92**	60.40	6.03	**64.08**	**4.71**
10	84.97	25.21	**61.40**	**5.71**	63.27	4.74

Table 6.7: Authentication results of an FRF verifier (in %) for different number of trees averaged over all 80, 50 and 49 users for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* broadcast events, respectively.

	USER_PRESENT		*PACKAGE_REMOVED*		*PACKAGE_ADDED*	
#trees	TAR	FAR	TAR	FAR	TAR	FAR
2	29.19	23.27	25.00	13.70	21.84	13.42
4	62.40	32.88	36.00	15.27	32.29	10.88
8	71.69	24.94	29.40	10.56	28.77	10.65
16	80.97	24.03	37.00	6.83	36.12	9.70
32	82.72	24.97	45.40	4.37	44.08	5.18
64	83.81	24.51	51.80	3.66	48.98	5.07
128	84.19	24.31	52.60	3.65	50.00	3.04
256	84.47	24.31	**53.00**	**5.39**	50.61	3.07
512	84.63	24.26	52.40	5.37	51.42	3.07
1024	**84.78**	**24.26**	52.60	5.48	**51.84**	**3.03**

6.5 Discussion

We present a completely unobtrusive phone's micro-movement based one-shot-cum-continuous smartphone user authentication scheme - *ACTIVEAUTH* which grants access to only a genuine user, on one hand, and ensures an authenticated session, on the other. *ACTIVEAUTH* starts monitoring the user right from the time of unlocking and keeps tracking the package installation and/or uninstallation throughout the session.

As a first step, we analyzed the phone's micro-movement data generated right after the triggering of the three broadcast receivers, namely, *USER_PRESENT*,*PACKAGE_REMOVED*, and *PACKAGE_ADDED*. We exploit only these broadcast events because they don't require any user permission, interaction, or cooperation. For the intended (cooperative) users *ACTIVEAUTH* can be equipped with multiple broadcast receivers and almost every user action (e.g., sending SMS, opening a banking app, etc.) can be authenticated.

In total 123 user downloaded the application, however, for various reasons, we could utilize 80, 50, and 49 users data associated with *USER_PRESENT*, *PACKAGE_REMOVED*, and *PACKAGE_ADDED* broadcast receivers, respectively. We take into account 40 samples per user for *USER_PRESENT* event, and 10 samples per users for the other two broadcast events. For us, number of users are more important than number of observation per sample, so we set a threshold of just 10 samples per user to accomodate more users (for *PACKAGE_REMOVED* and *PACKAGE_ADDED*), similar to TDAS dataset[111] - their collected keystroke based dataset contains just 10 samples from each of 150 users.

All the chosen verifiers showed better accuracy based on CSE features as compared IGAE features and on full feature set (Table 6.5). The reason is because most of the features are found redundant and non-productive (as shown in the Figure 6.4). It is worth noticing that with few CSE-based features, i.e., 13, 11 and 12 for *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* (Table 6.4), respectively, we observed significant increase in the obtained accuracy. Additionally, the computational cost of these shorter feature vectors will be very less as compared to those with full features.

This accuracy is further improved by applying parameter optimization. We evaluate the performance of *ACTIVEAUTH* over the range of hidden layers (1 : 10), number of trees (2:1024) and regularization parameters (0.0000625:0.032). We show that the performance can further be improved by applying the verifier's parameter optimization. With different regularization parameters, Gauss_DD is the most accurate verifier and classified with an accuracy of 82.24% for *USER_PRESENT*, 92% for *PACKAGE_REMOVED*, and 93.88% for *PACKAGE_ADDED* data.

Among all the chosen verifiers, Gauss_DD outperformed the other two verifiers with a final TAR of 84.06% (at 19.57% FAR), 98% (at 14% FAR), and 93.88% (at 9.1% FAR) for *USER_PRESENT*, *PACKAGE_REMOVED*, and *PACKAGE_ADDED*, broadcast events, respectively. We consider 80, 50, 49 users sufficient enough to prove the initial intuition.

As a future work, we will prototype a proof-of-concept app and evaluate it in terms of its performance and usability. The app will continue collecting the corresponding data from the above-mentioned broadcast receivers in daily usage and after reaching a best number (like 50 - 100 observations), it will notify the user about the availability of this behavioral modality for authentication.

6.6 Chapter Summary

In this chapter, we presented a fully unobtrusive one-shot-cum-continuous smartphone user authentication scheme. The proposed scheme monitors the entire user session right from the start (at login) and keeps tracking the user interactions, i.e., the addition/removal of an application package. Our scheme collects data from multiple built-in smartphone sensors after the OS notifies the user presence through *USER_PRESENT*, package installation through *PACKAGE_ADDED* and package uninstallation through *PACKAGE_REMOVAL* broadcasts. The scheme uses state-of-the-art verifiers to authenticate the user, and grants access to the genuine user. This property of passive authentication validates *ACTIVEAUTH* as an user-friendly authentication scheme. Similarly by exploiting multiple built-in sensors (avoiding additional hardware) *ACTIVEAUTH* can be enabled on any of off-the-shelf smartphone available in the market today.

Preliminary results of *ACTIVEAUTH* validate the effectiveness of the proposed scheme. Though our obtained verification results are based on a relatively smaller dataset, still showing the feasibility of the scheme for a non-intrusive and user-friendly one-shot-cum-continuous smartphone authentication system.

Chapter 7

ITSME: A Multi-Modal System for Transparent User Authentication on Smartphones

In this chapter, We propose a new tri-modal behavioral biometric that uses features collected while the user slide-unlocks the smartphone to answer a call. In particular, we use the slide swipe, the arm movement in bringing the phone close to the ear, and voice recognition to implement our behavioral biometric. We implemented the method on a real phone and we present a controlled user study among 26 participants in multiple scenarioâĂŹs to evaluate our prototype. We show that for each tested modality the Bayesian network classifier outperforms other classifiers (Random Forest algorithm and Sequential Minimal Optimization). The multi-modal system using slide and pickup features improved the unimodal result by a factor two, with a FAR of 11.01% and a FRR of 4.12%. The final HTER was 7.57%.

This chapter is based on our published work in [29]: Attaullah Buriro, Bruno Crispo, Filippo Del Frari, Jeffrey Klardie, and Konrad Wrona. ITSME: Multi-modal and unobtrusive behavioral user authentication for smartphones, in Proceedings of the International Conference on Passwords, pp. 45-61, Springer, 2015.

7.1 Introduction

Unimodal systems use information from a single source, as such they had to deal with a range of problems like noisy data, spoof attacks and unacceptable high error-rates. Some of these issues can be addressed by combining multiple sources of information [112]. Due to the presence of multiple (mostly) independent features, the performance is expected to increase [113].

Using biometrics authentication for smartphone users faces two important challenges.

First, users may use the phone in different situations and contexts (i.e. while walking, sit on a chair, standing up, in the dark, etc.). Thus, any realistic solution should accommodate the possibility that data acquisition may fail or that a particular feature might be temporarily unavailable. Second, the solution must require as small effort as possible to users. Studies suggest that usability issues are a major driver of users' adoption decisions [20].

To partially address these challenges this chapter presents a novel multi-modal biometric system for smartphone users authentication. The system uses slide-unlock features, pickup movements and voice features while placing or answering a call. Being multi-modal the solution aims at robustness, such that the users can still be authenticated even if some of the modalities fail.

To address the problem of usability, our authentication scheme requires *zero* effort to the users. The users are not required to perform any action for the sole purpose of authentication. In fact, entering a password or PIN is more noticeable. Last but not least, our system can be implemented on most of the smartphones available on the market today.

7.1.1 Contributions

The main contributions of this paper are:

- The proposal of a novel and fully unobtrusive tri-modal behavioral biometric user authentication solution, based on: *slide* - how a user slide-unlocks, *pickup* - how a user brings her phone to her ear for call making/answering, and *voice* recognition.
- Experimental validation of *ITSME* in different situations.
- The collection and sharing of data from multiple sensors in multiple user situations from 26 users.

7.2 Related Work

This section reports related work that specifically take mobile devices into consideration. A wider survey of biometric authentication in general can be found in [2, 114] and [30].

7.2.1 Unimodal Systems

In [115], Frank et al. consider touch operations for continuous authentication where a single type of operations are used (strokes or slides). An EER of 13% has been reported for one single stroke, and 2% to 3% for 11 subsequent strokes. In [56], a user is authenticated not only on the password pattern provided, but also the way she performs that input. A lab study and a long-term study provide evidence that it is possible to distinguish users

and to improve the security of password patterns on even simple screen unlocks. The accuracy rate of the simple unlock is 57% at best (two-finger vertical unlock), while the accuracy of the password patterns is around 77%. In [57], Angulo et al. explored the same approach for improving password-patterns with biometrics. Using a Random Forest classifier an EER of approximately 10.4% is achieved.

Sae-Bae et al. [58] presented a multi-touch gesture-based authentication technique. In such approach, a classifier, which uses pattern recognition techniques, classifies movements characteristics of the center of the palm and fingertips. An average EER of 10% with single gestures was achieved, with improvements up to 5% EER when combining multiple gestures in a sequence.

In [97] Derawi et al. authenticate users based on gait recognition using accelerometers available in any modern mobile device. Using a low end phone (the Google G1 phone containing the AK8976A embedded accelerometer sensor) an EER of 20% is reached.

Tao et al. [116] implement a fast face detection and registration method based on a Viola-Jones detector [117]. A face-authentication method based on subspace metrics is developed. Experiments using a standard mobile camera showed that the method is effective with an EER of 1.2%.

7.2.2 Multimodal Systems

In [118], Saevanee et al. used SMS texting activities and messages in a multimodal authentication system. Keystroke dynamics and linguistic profiling was used to discriminate users with error rates of 20%, 20% and 22%, respectively. A fusion of these three led to an overall EER of 8%.

Aronowitz et al. [119] introduced a new biometric modality called "`chirography`" which is based on user's writing on multi-touch screens using their fingers. By fusing this with face and voice features, an EER of 0.1% is reached in an office environment, and 0.5% in noisy environments.

In [120] Ferrer et al. introduced a multimodal biometric identification system that is based on the combination of geometrical, palm and fingerprint features of the user's hand.

In [121] a multimodal authentication approach is presented by Kim et al., using teeth and voice data acquired using mobile devices. The individual matching scores obtained from these biometric traits are combined using a weighted-summation operation. An EER of 2.13% was reported.

In [122], McCool et al. introduced a fully automatic bi-modal face and speaker system. A Nokia `N900` was used during tests and EER results of 13.3% and 11.9% for female and male trials respectively have been reported for the fused score. This is a 25% performance improvement for the female trials, and 35% improvement for male trials.

7.3 Approach

In this section, we explain the technology and building blocks we used to build our solution.

7.3.1 Intuition Assessment

In [28] Conti et al. introduce a new biometric measure to authenticate smartphone users; the movement a user performs when answering (or placing) a phone call. Authors show that this movement is unique enough for authentication. Interested readers are referred to this study [28] for intuition assessment.

7.3.2 Our Solution

Several experiments with a prototype based on this study [28] in a controlled environment have shown that the method is effective and that the performance is comparable to that of other transparent authentication methods, based on face or voice modalities. These experiments also highlighted an issue with the data acquisition process, due to the variability in determining the end of the arm movement. To address this issue without compromising the unobtrusive nature of the initial idea we extended the solution as follows:

When placing or answering a phone call, three common steps have to be taken: 1) the user must unlock her phone, 2) bring it to her ear, and 3) speak to the microphone. Our multi-modal authentication solution uses features from all three steps to determine whether or not the current user is genuine, or if she is an impostor.

The complete system consists of four parts: slide movement recognition, pickup movement recognition, voice recognition and fusion. The data features are described in this section, while the next section describes the actual classification framework including fusion.

7.3.3 Considered Sensors

We considered three built-in smartphone sensors, namely, accelerometer, orientation and gyroscope for movement and MIC for voice recording, respectively.

Voice recognition has been a well tested and evaluated modality. Therefore we did not create a new method for this modality, but decided to use an existing open source implementation that worked with our mobile environment. Creating the voice models consists of several steps:

1. An audio sample is recorded for $2500ms$ at a sample rate of 8 kHz using 16 bits per sample with one channel. The resulting pulse-code modulation (PCM) data is stored in a temporary WAV file on the device.

2. Using the recorded voice sample, we calculate the Mel-Frequency Cepstral Coefficients (MFCCs) [123] and store them in a feature vector. MFCCs have been very popular in the realm of speech recognition due to its ability to represent the speech amplitude spectrum in a compact form [124]. Creating MFCCs is done by 1) converting the waveform into frames, 2) take the discrete Fourier transform, 3) take the Log of the amplitude spectrum, 4) Mel-scaling and smoothing, and 5) applying the discrete cosine transform.

3. Apply KMeans clustering to partition the dataset into k clusters where each observation belongs to the cluster with the nearest mean.

4. The MFCC features are then used as data instances that we use to create models for our classifiers.

7.3.4 Considered Classifiers

We performed verification with three different verifiers, i.e., 1-class BayesNET (BN) classifier, 1-class RF and 1-class Sequential Minimal Optimization (SMO)- a WEKA version of support vector machine (SVM). We chose these classifiers because they were shown to be very efficient in previous behavioral-based work [27, 31]

We imported WEKA library in our project and implemented our prototypes on the smartphone.

7.4 Experimental Analysis

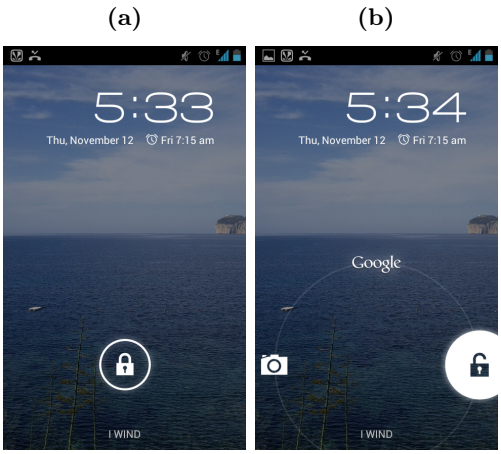
7.4.1 Setup

We conducted a controlled user study to test the effectiveness of our mechanism. We recruited 26 participants of which 16 were male, and 3 operated their phone using their left hand. All of them were familiar with the slide-to-unlock pattern. Ages of our volunteers were ranging from 14 to 55. 2 participants were $14-19$, 12 were $20-29$, 7 were $30-39$, 1 was $40-49$ and 4 were 50 or older.

We created an Android application that targets SDK version 4.4 (*Kitkat*) and minimally requires version 4.0.3 (*Ice Cream Sandwich*). We implemented both the training phase and the classification phase using WEKA 3.7 on an android smartphone. The training module allows the user to anonymously record slide movements, pickup movements and voice samples, which are sent to a central server. The classification module was implemented as a proof of concept and to analyze the performance on mobile phones.

A central server running on the Amazon cloud platform collected the training features in a database. A local running Java application (using Java 1.7) using the same classification module as implemented in Android was then used to test the robustness of the

Figure 7.1: Android slide lock: (a) the default state, (b) the state when a user drags the knob towards the circular boundary.



system. We used a Google Nexus 4 device by LG running Android 5.1 during the study. This device has a 4.7 inch screen, a Qualcomm APQ8064 Snapdragon 1.5 GHz Quad-core processor, 2 GB RAM, an accelerometer, a gyroscope and a proximity meter.

In each session, we first explained the purpose of the study to the participants and asked them if we could use their data anonymously, and noted their age and gender. After that we moved to the actual trials. Each user was required to collect at least 20 slide samples, 20 pickup samples and 10 voice samples. Samples that were distorted in any way could be removed by the user.

For the slide and pickup movements, we instructed the participants to first do five movements while sitting or standing still and after that five while walking around. Then the user was asked to open a news app and read the fifteenth headline, which required the user to count while scrolling to the headlines. This usually confused users, and many had to recount from the top because they tried to wrap their head around the purpose of this task, and lost count. The goal of this distraction task was to minimize the learning effect that can occur when doing the same movement many times in quick succession. After the user read the article, she was again requested to record five movements while sitting, and five movements while walking.

7.4.2 Data Collection

We use the default Android slide lock as depicted in Figure 7.1. The center knob can be dragged towards any direction. When the user drags the knob and then release it, at least as far as the circular boundary (slightly visible in the right image in Figure 7.1), the phone will be unlocked. If the knob is released before reaching the boundary, the phone

stays locked.
During the training phase a pickup event starts when the user clicks the start button, and ends automatically when the phone is at the user's ear (detected by the proximity detector). When used in combination with the other two modalities (e.g., during authentication), the sample starts when the slide unlock ends, and also finishes when the phone reaches the user's ear.
The Android system continuously delivers SensorEvents[1] to an event listener. As we use three sensor (accelerometer, gyroscope, orientation sensor), a delivered event can be produced by one of the sensors. Every time we receive a new event for any of the sensors, we extract the x, y and z values, and store them.
For the voice sample recordings, we requested the user to simply speak into the microphone as if they were answering a phone call, but to make sure to use a relatively lengthy sentence to fill the 2.5 seconds of recording. Most users used a sentence similar to *Hello, this is John Doe. Who am I speaking to?*. An audio sample is recorded for $2500ms$ at a sample rate of 8 kHz using 16 bits per sample with one channel. The resulting pulse-code modulation (PCM) data is stored in a temporary WAV file on the device.

7.4.3 Feature Extraction

7.4.3.1 Slide

A slide sample starts when the user touches the knob for the first time, and ends when the knob is released (e.g., the user stops the touch event). One slide is a path encoded as a sequence of vectors $(t_n, x_n, y_n, p_n, s_n)$. Only complete samples (samples that would unlock the phone in the original non-biometric implementation) are considered, others are simply discarded.

During the slide event the features in Table 7.1 are recorded at a average sampling rate of 150 Hz. From the given MotionEvent we extract multiple features. The *time offset* (t_n) indicates the offset since the start of the touch event in milliseconds.

Table 7.1: Slide features.

Feature	Unit
Time offset	ms
X-position	px
Y-position	px
Pressure	Normalized value between 0 and 1
Size	Normalized value between 0 and 1

The *x- and y-position* (x_n, y_n) are measured in pixels and indicate the exact position of the knob (controlled by the users touch) on the screen. Over time these coordinates create a path

[1] http://developer.android.com/reference/android/hardware/SensorEvent.html

from the initial position of the knob towards the boundary of the circle, indicating exactly how the user moved the knob.

The *touch pressure* (p_n) of the touch event indicates the approximate pressure applied to the surface of the screen. The value is normalized to a range from 0 (no pressure at all) to 1 (normal pressure), but values higher than 1 may be generated depending on the calibration of the input device.

The *size* (s_n) is a scaled value of the approximate size of the area of the screen being touched. The actual value in pixels corresponding to the touch is normalized with the device specific range of values and scaled to a value between 0 and 1.

7.4.3.2 Pickup

During the pickup event the features in Table 7.2 are extracted at an average sampling rate of 190 Hz. The *time offset* (t_n) indicates the offset since the start of the pickup event in milliseconds. One pickup movement is encoded as a sequence of vectors ($acc_n^x, acc_n^y, acc_n^z, gyro_n^x, gyro_n^y, gyro_n^z, rot_n^x, rot_n^y, rot_n^z, t_n$).

Table 7.2: Pickup features.

	Features			Units
$1-3$	X-acceleration	Y-acceleration	Z-acceleration	m/s^2
$4-6$	X-gyroscope	Y-gyroscope	Z-gyroscope	rad/s
$7-9$	X-orientation	Y-Orientation	Z-Orientation	rad
10	Time offset			ms

7.4.3.3 Voice

Using the recorded voice sample, we calculate the Mel-frequency Cepstral Coefficients (MFCCs) [123] and store them in a feature vector. MFCCs have been very popular in the realm of speech recognition due to its ability to represent the speech amplitude spectrum in a compact form [124]. The MFCC features are then used as data instances that we use to create models for our classifiers.

7.4.4 Data Fusion

In our multi-modal mechanism, we use multiple biometric traits (slide movement, pickup movement and voice) which need to be fused to output one single decision: accept or

reject. We fused these modalities at match-score level. However, because each modality performed differently, we give each modality a weight, based on it's unimodal performance.

Consider three modalities x, y and z, having an error rate (er) of 0.1, 0.2 and 0.3 respectively. Obviously, modality x is much better than y and z, and should therefore have a higher weight. For each classifier c we can calculate a success index. The success index indicates how much the classifier contributes to the sum $1 - er(c)$ for each classifier c.

$$index(c) = 1 - \frac{er(c)}{\sum_{i=1}^{n} er(i)} \tag{7.1}$$

The eventual weight can then be calculated using:

$$weight(c) = \frac{index(c)}{\sum_{i=1}^{n} index(i)} \tag{7.2}$$

Filling in the values for three modalities x, y and z, they would get weights of 0.42, 0.33 and 0.25, respectively. Better modalities get higher weights.

7.4.5 Analysis

During the training phase, we only have training data available for a single instance class; the genuine user (the *target* class). At prediction time new instances with unknown class labels will have to be classified as either the *target* class or *unknown*. To handle this type of learning problem, typically called 1-class classification, we wrap each classifier in a 1-class classifier[2].

7.4.5.1 Decision Making

To measure the performance of the classifiers, we use the cross-validation method. The dataset is randomized and then split into k folds of equal size. In each iteration, one fold is used for testing, and the other $k - 1$ folds are used for training the classifier. We use $k = n$, meaning we apply leave-one-out cross-validation. The test results are averaged over all folds, which give the cross-validation estimate of the accuracy. This method is useful because we are dealing with small datasets and the idea is to test each sample. Using cross-validation we utilize the greatest amount of training data from the dataset.

[2]http://weka.sourceforge.net/packageMetaData/oneClassClassifier/

When evaluating the performance of a biometric system, multiple criteria should be considered [41]. Biometric authentication systems make decisions based on the following decision function:

$$f(x) = \begin{cases} accept, & \text{if } c(I, \mathrm{x}) \geq \Delta \\ reject, & \text{otherwise} \end{cases} \tag{7.3}$$

where $c(I, f)$ is the output of the underlying classifier c that indicates how certain it is that the claimed identity I is correct based on the given dataset (features) x. The threshold Δ defines when an identity claim is accepted or rejected. Access to the system is granted if the score is greater than or equal to the threshold, and rejected otherwise.

7.5 Parameters and Attributes Selection

Before we show any results, we first need to identify the exact data and models under test. During the research we did extensive experiments to find the optimal setup. This section will describe the results of these intermediary tests which will lead us to the best performing combination of parameters and attributes. The actual performance of the best classifier will be discussed and evaluated in the next section.

These tests have been carried out on a random subset (length: 10) of the participants in the user test. For each configuration considered, we calculated the equal EER based on all samples of the genuine user, and 10 random samples per other (non-genuine) user.

7.5.1 Parameters

This section will outline the steps taken to find the best set of parameters per classifier per modality. For each modality, we use all features as described in Section 7.4.3, and do a grid search to find the best performing set of parameters. We also record the average model generation time so we can filter out configurations that would take too long on mobile phones.

Below we will discuss the parameters per classifier, while the results are presented at the end of this subsection.

7.5.1.1 BayesNet Classifier

For the BayesNET classifier, we tested each combination of parameter values shown in Table 7.3. Preliminary tests have shown that the use of a *Random order* or *AD trees* had almost no effect on the outcome of the classifier. Therefore we set them to false and did not consider them in the grid search.

Table 7.3: Parameters considered in BN grid search.

Parameter	Considered values
Score type	$\{Bayes, BDeu, MDL, Entropy, AIC\}$
Max parents	$\{0, 1, 3, 5\}$
Alpha (α)	$\{0, 0.25, 0.50, 0.75\}$
Naive bayes	$\{yes, no\}$
Markov blanket	$\{yes, no\}$
Random order	$\{no\}$
AD trees	$\{no\}$

Appendices A.1, A.2 and A.3 show the top 20 parameter configurations for each modality. Note that for the pickup modality, because of time limitations, we only tested a maximum of 3 and 5 parents.

7.5.1.2 SMO Classifier

The effectiveness of SMO mainly depends on the kernel, the kernel's parameters, and the parameter C. Often a Gaussian kernel is used [125], which only takes one parameter *gamma* (γ). We will use the same approach and do a grid search with exponentially growing sequences of C and γ to find the best combination of these parameters.

Table 7.4: Parameters considered in the SMO grid search.

Parameter	Considered values
C	$\{2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}\}$
Gamma (γ)	$\{2^{-8}, 2^{-4}, 2^{0}, 2^{1}, 2^{2}\}$
Epsilon (ϵ)	$\{1E-12\}$
Num folds	$\{-1\}$

Table 7.4 shows the parameter values we considered in our search. Appendices A.4, show the top 20 parameter configuration for slide modality only.

We tested each possible combination listed in the Table 7.4. Note that we did not finish the SMO test for the pickup and voice modalities. The first test result took approximately 144 seconds of computation time per model. Considering each parameter configuration test computes 236 models, and we tested 25 combinations, it would take almost 10 days to finish. Such long computation times are unacceptable on mobile phones.

7.5.1.3 RF Classifier

The search for the optimal parameter set in the RF classifier is rather simple, as only the number of trees used is of major influence on the outcome. However, picking the right number of trees is not necessarily a trivial task. Research by Oshiro et al. [110] has shown that a larger number of trees in a forest does not always have a significant performance

gain, while it does increase the computational cost.

Experiments by Oshiro et al. [110] concluded that a range of 64 to 128 trees could provide a good balance between performance, processing time and memory usage. As such, we used these numbers as a starting point and tested 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024 number of trees.

Appendices A.5, A.6 and A.7 show the parameter configurations for each different number of trees, for each modality.

Table 7.5: Best classifier per modality.

	Slide		Pickup		Voice	
Classifier	Comp. Time	EER	Comp. Time	EER	Comp. Time	EER
BayesNET	64	0.1242	762	0.2045	205	0.2681
RF	4453	0.1434	13988	0.2083	4402	0.2452
SMO	8433	0.1864	~144000	-	548	0.2709

Table 7.5 gives an overview of the best performing classifiers for each modality. The parameter tests show that the BayesNET classifier yield the best results overall. Only with the voice modality the RF classifier yields slightly better results. However, the BayesNET is much faster.

Further tests in this chapter will be done only with the best performing classifiers. From this point on when talk about the classifier, we mean the BayesNET classifier, with it's parameters configured as shown in Table 7.6, based on the modality at discussion.

Table 7.6: Parameter configuration per modality

Modality	Naive Bayes	Markov blanket	Max parents	Score type	Alpha
Slide	T	T	5	Entropy	0.25
Pickup	T	F	3	Bayes	0
Voice	F	F	5	Entropy	0

7.5.2 Attribute Selection

Besides the parameters used to configure the classifiers, another aspect of high influence is the data attributes that are being used. To find out the best setup, again we perform a grid search.

Appendix A.8 shows the top 15 attribute configurations for the slide modality. The classifier performs best when all attributes are being used: *(x, y, pressure, size, offset)* (see first row of the Appendix A.8). Appendix A.9 shows the best performing pickup

attribute settings. This time, we see the best results when a subset of the attributes are being used: *(accX, accY, accZ, gyroX, gyroY, gyroZ, rotY, rotZ, offset)*. So, note that *rotX* has been excluded from further analysis.

We will use the BayesNET classifier configured per modality as described in the previous section. From here on, when we discuss the classifier, it will be the BayesNET classifier using all attributes.

7.6 Results

The results we present here are based on the user data we collected during the controlled users tests, fed into the classifiers with their parameters configured as described in the Section 7.5.1. For each user this gives us a certainty number (higher means more similar to the reference model) for both genuine and impostor samples.

It is important to note that when testing a classifier for user u, we use all samples from all other users to do our impostor tests. By doing so, we have much more impostor samples than genuine samples, leaving the FRR much more sensitive to deviations than the FAR.

Given the data from the user we can find the optimal threshold Δ^*_α. The optimal threshold is the threshold for which the WER rate is at its minimum (see Equation 2.5).

7.6.1 Unimodal Systems

7.6.1.1 Slide

We tried different values α to find the optimum threshold of given $\alpha = 0.4$, we found that the optimal threshold $\Delta^*_\alpha = 49$. Re-running the tests with this threshold gives us a FAR of 22.28% and a FRR of 4.84%.

The HTER (defined in Equation 2.6) can now easily be computed:

$$HTER(49) = \frac{22.28\% + 4.84\%}{2} = \textit{13.56\%} \tag{7.4}$$

7.6.1.2 Pickup

The optimal threshold $\Delta^*_\alpha = 42$. Running the tests with this threshold gives us a FAR of 26.69% and a FRR of 6.19%.

$$HTER(42) = \frac{26.69\% + 6.19\%}{2} = \textit{16.44\%} \tag{7.5}$$

7.6.1.3 Voice

The optimal threshold $\Delta_\alpha^* = 85$. Running the tests with this threshold gives us a FAR of 63.92% and a FRR of 12.69%.

$$HTER(85) = \frac{63.92\% + 12.69\%}{2} = \textit{38.30\%} \tag{7.6}$$

It is evident that slide and pickup modalities are better than voice modality. Still, we are using it here to show how the use of multi-modal biometric authentication can improve a unimodal authentication system.

7.6.2 Multi-modal Systems

7.6.2.1 Slide+Pickup Modalities

As described in Section 7.4.4 we use a match-score level fusion method, using weights for each classifier output. We calculate the weight using Equation 7.2. In the previous subsection, we have seen that the slide and pickup classifiers have a HTER of 13.56% and 16.44% respectively. Filling in the equation this gives us a weight of 0.55 for slide and 0.45 for pickup.

The optimal threshold $\Delta_\alpha^* = 55$. Re-running the tests with this threshold gives us a FAR of 11.01% and a FRR of 4.12%. Calculating the HTER gives us:

$$HTER(55) = \frac{11.01\% + 4.12\%}{2} = \textit{7.57\%} \tag{7.7}$$

Comparing the slide and pickup modalities individually with this multimodal system, we can see that the latter performs almost twice as good.

7.6.2.2 Slide+Pickup+Voice Modalities

We have seen that the slide, pickup and voice classifiers have HTERs of 13.56%, 16.44% and 38.30% respectively. Using Equation 7.2 this results in weights 0.40 (slide), 0.38 (pickup) and 0.22 (voice).

The optimal threshold $\Delta_\alpha^* = 62$. Running the tests with this threshold gives us a FAR of 10.72% and a FRR of 3.93%. Calculating the HTER gives us:

$$HTER(62) = \frac{10.28\% + 3.93\%}{2} = \textit{7.33\%} \tag{7.8}$$

A quick comparison shows that adding voice modality to the multimodal system using slide and pickup features does not improve the results significantly but still better (HTER 7.33% Vs HTER 7.57%).

7.7 Discussion

The results show that the slide modality is better than the pickup modality. The main cause for this observation is that the pickup modality is much more sensitive to the kind of activity the user performs while unlocking her phone. Because the rotation, gyroscope and acceleration of the device are the main features of the modality, the user's activity while unlocking has major influence on the classifier outcome: walking, running, standing in a crowded bus; they all have different impact on the motion sensors of the device.

The slide modality on the other hand does not use motion sensors but rather uses touchscreen. Touchscreen determines finger position, pressure and size on a screen which are much less influenced by external factors, making the modality more robust in a range of different scenario's.

When combining the slide and pickup modalities, we can see that the FAR improves significantly.

The voice modality is obviously not good enough (based on our experiments) and may not be deployed in real world because of higher error rates - FAR of 63.92% and FRR of 12.69%. The reason(s) for worst voice results may be due to the low quality of the open source library and by the fact we applied only basic clustering mechanisms. Still, the fusion of all three modalities yielded better results.

System like ours are suitable for risk-based authentication scenarios (e.g., mobile banking applications), where security may need to be traded for availability dynamically and adaptively.

This research can be extended in multiple directions. To validate the results presented here a larger user study should be conducted. The impact of situations, context and environment may have on this type of biometrics need to be investigated further.

7.8 Chapter Summary

In this section, we proposed a new multi-modal biometric system for smartphone user authentication that focuses on usability. The system uses features collected during a slide-unlock movement on a smartphone. We use finger position, pressure, size and time offset to generate a model and classify future slide movements. We shown how the fusion of unimodal systems to multi-modal ones, using slide, pickup and voice modalities, can significantly improve the authentication performance.

We have applied three different classifiers, i.e., BN, RF and SVM. BN classifier outperformed the other classifiers in terms of error rates and computation time.

From the three unimodal modalities we tested (slide, pickup and voice); the slide modality performed best with a FAR of 22.28% and a FRR of 4.84%, resulting in a HTER of 13.56%. The pickup modality performed slightly worse, with an FAR and FRR

of 26.69% and 6.19% respectively, and an HTER of 16.44%. However, with their fusion, we were able to achieve much improved performance (by a factor of two). A FAR of 11.01% and a FRR of 4.12% were reached, resulting in a HTER of 7.57%.

The voice based model performed much worse as the open source library we used was simply not good enough. However, we have shown the potential improvement of a multi-modal system using slide, pickup and voice modalities.

Chapter 8

Conclusion and Future Work

In this dissertation, we have addressed the problem of user authentication on smartphones by proposing, efficient, robust, user-friendly and hardware-friendly, behavioral biometric based solutions as a replacement to existing cumbersome and annoying authentication mechanisms. In particular, we have identified a novel human behavior - the *"Hold"* behavior, by which users can be profiled and verified, transparently without any hassle. It becomes extremely difficult to mimic the *"Hold"* behavior because of the natural differences in the human body structure. We have reported its efficacy in both unimodal (Chapters 5 & 6) and bi-modal (Chapters 3 & 4) systems. For example, in the Chapter 5, we have proposed a fully *"Unobtrusive User Authentication"* mechanism, based on the profiling of user's hand micro-movements after the occurrence of an unlock event. Based on the collected user's hand micro-movements with the help of state-of-the-art machine learning classifier, our scheme assesses whether or not the smartphone is activated by the legitimate user. Similarly, In Chapter 6, we proposed *ACTIVEAUTH* - an extended version of our earlier proposed approach (Chapter 5). This approach is equally useful both for one-shot and continuous authentication scheme. *ACTIVEAUTH* is based on the profiling of user's hand micro-movements after the occurrence of *USER_PRESENT*, *PACKAGE_REMOVED* and *PACKAGE_ADDED* broadcast receivers. Both methods do not involve any PIN, password or token or any other remembered action and require *zero* user effort, thus, they are completely transparent and applicable on the smartphones with and without any authentication mechanism enabled. This property of passive authentication validates the two mechanism (5 & 6) as the user-friendly authentication schemes.

We have shown that the accuracy of multi-modal system increases to a significant extent by adding this modality to the existing touch-based approaches. For example, in Chapter 3, we have proposed a user authentication schemes based on two human behaviors, i.e., (i) how a user types a 4-*digit free-text* PIN, password on the touchscreen, and (ii) the phone movements while doing so. Similar to the Chapter5, we profiled the

user hand's movement with the available built-in sensors, however the profiling time was short - time required to enter 4-*digit* most preferred PIN, password. The participants were allowed to use any combination of 4-*digit* number and alphabets. The addition of phone-hold modality and the choice of any combination of text differentiates *"Touchstroke"* from the classical keystroke dynamics. The users of *"Touchstroke"* were found to be quite comfortable while using this authentication mechanism because it transparently collects the data of hold-behavior and provides the flexibility of entering any *4-digit* text. Similarly, Chapter 4 presents a variant of *"Touchstroke"* and is based on two human behaviors- how a user holds her phone in one hand (phone-movement) and how she writes her name on the touchscreen. We profiled the phone micro-movements using smartphone sensors (like *"Touchstroke" 3*) and register touch behavior using the pressed touch-points collected from the touchscreen. Our proposed *"Hold & Sign"* method does not take into account the image (because the image can be copied and mimicked [51]). We do acknowledge that typing a PIN is easier than writing something on the touchscreen, however, the PINs can be forgotten, whereas the user always remembers their name. It becomes extremely difficult to launch shoulder surfing and smudge attacks as compared to steal the PINs and password. In *"Hold & Sign"*, even if an attacker can see what is being written on the touchscreen, the access can still be denied because a would-be adversary cannot mimic the phone-movements of the legitimate user.

We have proposed multiple schemes for smartphone user authentication, and now in the process of finalizing the other implementations along-with their security and usability evaluation (for Android devices). We have reported in this thesis an evaluation of one of our proposed mechanisms (*Hold & Sign)* in terms of security, robustness and usability. Obtained results indicate a positive sign of user acceptability.

What makes our solutions unique and better is their minimal or no user effort requirement - our solutions authenticate their users with either minimal or without their explicit user cooperation. In addition, all of our solutions exploit the existing hardware (avoiding additional hardware requirement), and hence can be implemented on most of the smartphone available in the market today.

8.1 Future Research Dimensions

Despite great progress in mobile biometrics including the work presented in this thesis, there exist several challenging issues to be addressed yet. The research work presented in this dissertation can be extended in the following dimensions:

8.1.1 Prototyping Proof-of-Concept Applications

We are in the process of finalizing and testing of our prototypes proof-of-concept applications based on our findings reported in this thesis (see Chapter 5,6, 3, and 7). Although, we introduced a activity-fusion strategy in *Hold & Sign* method (see Chapter 4), however, it was limited to just 3 activities. As the smartphone is supposed to be used in all the user positions, the final prototypes should be accurate enough in activity recognition to compare the query pattern with the pre-stored patterns related with that specific user activity. So, the final prototypes will take into account mechanism(s) to recognize the daily-based user situations (e.g., using JigSaw [91]) while authenticating.

8.1.1.1 Performance Analysis

We will evaluate our final proof-of-concept applications in terms of multiple performance related measures, i.e., (i) accuracy, (ii) sample acquisition time, (iii) authentication/decision time, (iv) incurred CPU and memory overheads, (v) power consumption, and (vi) deployment issues on smartphones. We will also try to solve the usability vs accuracy trade-off, i.e., how many patterns the user would like to register for training and based on this number which classifier works better. We will also evaluate the impact of choosing optimum number of samples over other performance related parameters.

8.1.1.2 Usability Analysis

We will test the usability of our prototype applications (both in in-lab and out-of-lab settings) using state-of-the-art usability analysis tools, e.g., Software Usability Scale (SUS)[1] etc. Our usability analysis will also comprise of some structured/semi-structured interviews to better evaluate the usability of our prototype applications/methods.

8.1.1.3 Adversarial and Security Analysis

Generally, the papers related to mobile biometrics report only the performance accuracy of the proposed authentication system, while ignoring the analysis against attacks. Hence, an study of their robustness against various attacks remain unexplored. More specifically, in such approaches, an impostor does not intentionally aims to fool the system without targeting any registered genuine user.

We will evaluate the robustness of all of our prototypes against the different attacks in different adversarial situations and report their accuracy against random, targeted and engineered attacks.

[1] https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

8.1.2 Permanency Analysis

The performance of any biometric-based authentication solution continuously varies due to observed within-person variations for various reasons related to environment and context. For example, the impact of being drunk on the accuracy of the system. Additionally, the accuracy is also affected due to aging and physical and/or mental health.

Research needs to be done to evaluate the impact of aging and the user's physical and/or mental conditions on our proposed schemes. Additionally, research on possible approach(es), to eliminate the effects of variations in health and age, need to be done in order to minimize the continuous deterioration in the performance of the the authentication solutions.

8.1.3 Analysis for Continuous Authentication

Continuous user authentication on smartphones, in general, and seemless or frictionless authentication, in particular, are clearly at initial stages and the proposed solutions are mainly based on users geographical location, device type, network, etc.

The proposed schemes can further be extended in terms of continuous user authentication.

8.2 Closing Remarks

In this dissertation, we have provided solutions for hassle-free and unobtrusive smartphone user authentication using behavioral biometrics. Instead of solving the problem with the existing approaches (using the existing datasets), we investigated and integrated novel behaviors (hence requiring fresh data) for the evaluation. Therefore, it is worth-mentioning that our obtained results are based on our collected datasets. In addition, all of our proposed solutions exploit built-in smartphone sensors hence avoiding the cost of any additional dedicated hardware.

Bibliography

[1] S. Trewin, C. Swart, L. Koved, J. Martino, K. Singh, and S. Ben-David, "Biometric authentication on a mobile device: a study of user effort, error and task disruption," in *Proceedings of the 28th Annual Computer Security Applications Conference*, pp. 159–168, ACM, 2012.

[2] A. Jain, A. A. Ross, and K. Nandakumar, *Introduction to biometrics*. Springer Science & Business Media, 2011.

[3] "Sensors overview." `https://developer.android.com/guide/topics/sensors/sensors_overview.html`, 2016.

[4] P. S. Teh, A. B. J. Teoh, and S. Yue, "A survey of keystroke dynamics biometrics," *The Scientific World Journal*, vol. 2013, 2013.

[5] A. Buriro, B. Crispo, F. Delfrari, and K. Wrona, "Hold and sign: A novel behavioral biometrics for smartphone user authentication," in *Security and Privacy Workshops (SPW), 2016 IEEE*, pp. 276–285, IEEE, 2016.

[6] A. TRUONG, "More iphones are sold than babies are born each day." `http://news.discovery.com/tech/gear-and-gadgets/more-iphones-are-sold-than-babies-are-born-each-day.htm`, 2012.

[7] "How smartphones are on the verge of taking over the world." `http://www.nydailynews.com/life-style/smartphones-world-article-1.1295927`, 2013.

[8] P. Cohen, "Macworld expo keynote live update,pc world." `http://www.macworld.com/article/1054764/liveupdate.html`, 2007.

[9] D. Morrill, "Announcing the android 1.0 sdk, release 1, google." `http://android-developers.blogspot.it/2008/09/announcing-android-10-sdk-release-1.html`, 2008.

[10] Khaley, "Introducing the symantec smartphone honey stick project." `http://www.symantec.com/connect/blogs/introducing-symantec-smartphone-honey-stick-project`, 2012.

[11] H. Spray, "Top 100 leaked nude celeb photos of all time." `http://www.hecklerspray.com/nude-celeb-pics/201164473.php`, 2016.

[12] M. Raza, M. Iqbal, M. Sharif, and W. Haider, "A survey of password attacks and comparative analysis on methods for secure authentication," *World Applied Sciences Journal*, vol. 19, no. 4, pp. 439–444, 2012.

[13] H. M. Wood, *The use of passwords for controlled access to computer resources*, vol. 500. US Department of Commerce, National Bureau of Standards, 1977.

[14] C. Theriault, "Survey says 70% password donâĂŹt protect mobiles: download free mobile toolkit." `http://nakedsecurity.sophos.com/2011/08/09/free-sophos-mobilesecurity-toolkit`, 2014.

[15] M. Jakobsson, E. Shi, P. Golle, and R. Chow, "Implicit authentication for mobile devices," in *Proceedings of the 4th USENIX conference on Hot topics in security*, pp. 9–9, USENIX Association, 2009.

[16] Android, "Google: Ice cream sandwich." `http://developer.android.com/about/versions/android-4.0-highlights.html`, 2011.

[17] Android, "Introducing: Smart lock." `https://get.google.com/smartlock/`, 2015.

[18] C. Velazco, "Apple touch id is a 500ppi fingerprint sensor built into the iphone 5s home button." `http://techcrunch.com/2013/09/10/apples-touch-id-a-500ppi-fingerprint-sensor,TechCrunch`, 2013.

[19] B. Alphonse, *Signaletic instructions including the theory and practice of anthropometrical identification*. Werner Company, 1896.

[20] C. Bhagavatula, B. Ur, K. Iacovino, S. M. Kywe, L. F. Cranor, and M. Savvides, "Biometric authentication on iphone and android: Usability, perceptions, and influences on adoption," *Proc. USEC*, 2015.

[21] A. De Luca, A. Hang, E. von Zezschwitz, and H. Hussmann, "I feel like i'm taking selfies all day!: Towards understanding biometric authentication on smartphones," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 1411–1414, ACM, 2015.

[22] Z. Akhtar, "Security of multimodal biometric systems against spoof attacks," *Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari, Italy*, vol. 6, 2012.

[23] Z. Akhtar, C. Micheloni, and G. L. Foresti, "Biometric liveness detection: challenges and research opportunities," *IEEE Security & Privacy*, vol. 13, no. 5, pp. 63–72, 2015.

[24] Z. Akhtar, C. Micheloni, C. Piciarelli, and G. L. Foresti, "Mobio_livdet: mobile biometric liveness detection," in *Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on*, pp. 187–192, IEEE, 2014.

[25] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, no. 4, pp. 367–397, 2002.

[26] N. L. Clarke and S. M. Furnell, "Authenticating mobile phone users using keystroke analysis," *International Journal of Information Security*, vol. 6, no. 1, pp. 1–14, 2007.

[27] A. Buriro, B. Crispo, F. Del Frari, and K. Wrona, "Touchstroke: smartphone user authentication based on touch-typing biometrics," in *International Conference on Image Analysis and Processing*, pp. 27–34, Springer, 2015.

[28] M. Conti, I. Zachia-Zlatea, and B. Crispo, "Mind how you answer me!: transparently authenticating the user of a smartphone when answering or placing a call," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pp. 249–259, ACM, 2011.

[29] A. Buriro, B. Crispo, F. Del Frari, J. Klardie, and K. Wrona, "Itsme: Multi-modal and unobtrusive behavioural user authentication for smartphones," in *proceedings of the 9th Conference on passwords (PASSWORDS 2015)*, pp. 45–61, Springer, 2016.

[30] R. V. Yampolskiy and V. Govindaraju, "Behavioural biometrics: a survey and classification," *International Journal of Biometrics*, vol. 1, no. 1, pp. 81–113, 2008.

[31] Z. Sitova, J. Sedenka, Q. Yang, G. Peng, G. Zhou, P. Gasti, and K. Balagani, "Hmog: A new biometric modality for continuous authentication of smartphone users," *arXiv preprint arXiv:1501.01199*, 2015.

[32] A. De Luca, A. Hang, E. von Zezschwitz, and H. Hussmann, "I Feel Like I'm Taking Selfies All Day!: Towards Understanding Biometric Authentication on Smartphones," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 1411–1414, 2015.

[33] P. A. Fahmi, E. Kodirov, D.-J. Choi, G.-S. Lee, A. M. F. Azli, and S. Sayeed, "Implicit authentication based on ear shape biometrics using smartphone camera during

a call," in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2272–2276, IEEE, 2012.

[34] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: A rule-based intrusion detection approach," *IEEE transactions on software engineering*, vol. 21, no. 3, pp. 181–199, 1995.

[35] F. Apap, A. Honig, S. Hershkop, E. Eskin, and S. Stolfo, "Detecting malicious software by monitoring anomalous windows registry accesses," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 36–53, Springer, 2002.

[36] A. Ross, K. Nandakumar, and A. K. Jain, "Introduction to multibiometrics," in *Handbook of Biometrics*, pp. 271–292, Springer, 2008.

[37] T. C. Clancy, N. Kiyavash, and D. J. Lin, "Secure smartcardbased fingerprint authentication," in *Proceedings of the 2003 ACM SIGMM workshop on Biometrics methods and applications*, pp. 45–52, ACM, 2003.

[38] C.-C. Lin, C.-C. Chang, D. Liang, and C.-H. Yang, "A new non-intrusive authentication method based on the orientation sensor for smartphone users," in *Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference on*, pp. 245–252, IEEE, 2012.

[39] "Common injuries." `http://www.vistalab.com/posture.asp`, 2014.

[40] D. Buschek, A. De Luca, and F. Alt, "Improving accuracy, applicability and usability of keystroke biometrics on mobile touchscreen devices," in *proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 1393–1402, ACM, 2015.

[41] N. Poh and S. Bengio, "Database, protocols and tools for evaluating score-level fusion algorithms in biometric authentication," *Pattern Recognition*, vol. 39, no. 2, pp. 223–233, 2006.

[42] C. Giuffrida, K. Majdanik, M. Conti, and H. Bos, "I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 92–111, Springer, 2014.

[43] X. Huang, G. Lund, and A. Sapeluk, "Development of a typing behaviour recognition mechanism on android," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 1342–1347, IEEE, 2012.

[44] H. Saevanee and P. Bhatarakosol, "User authentication using combination of behavioral biometrics over the touchpad acting like touch screen of mobile device," in *Computer and Electrical Engineering, 2008. ICCEE 2008. International Conference on*, pp. 82–86, IEEE, 2008.

[45] H. Saevanee and P. Bhattarakosol, "Authenticating user using keystroke dynamics and finger pressure," in *2009 6th IEEE Consumer Communications and Networking Conference*, pp. 1–2, IEEE, 2009.

[46] S. Zahid, M. Shahzad, S. A. Khayam, and M. Farooq, "Keystroke-based user identification on smart phones," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 224–243, Springer, 2009.

[47] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury, "Tapprints: your finger taps have fingerprints," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 323–336, ACM, 2012.

[48] 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*, pp. 41–50, ACM, 2012.

[49] N. Zheng, K. Bai, H. Huang, and H. Wang, "You are how you touch: User verification on smartphones via tapping behaviors," in *International Conference on Network Protocols (ICNP)*, pp. 221–232, IEEE, 2014.

[50] M. M. Dıaz and U. Ingeniero de Telecomunicación, "Dynamic signature verification for portable devices," 2008.

[51] J. Galbally, *Vulnerabilities and attack protection in security systems based on biometric recognition*. Javier Galbally, 2009.

[52] J. Mäntyjärvi, M. Lindholm, E. Vildjiounaite, S.-M. Mäkelä, and H. Ailisto, "Identifying users of portable devices from gait pattern with accelerometers," in *proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, vol. 2, pp. ii–973, IEEE, 2005.

[53] J. Zhu, P. Wu, X. Wang, and J. Zhang, "Sensec: Mobile security through passive sensing," in *Int. Conf. on Computing, Networking and Communications (ICNC)*, pp. 1128–1133, IEEE, 2013.

[54] L. Li, X. Zhao, and G. Xue, "Unobservable re-authentication for smartphones.," in *NDSS*, 2013.

[55] N. Sae-Bae, N. Memon, K. Isbister, and K. Ahmed, "Multitouch gesture-based authentication.," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 568–582, 2014.

[56] 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 SIGCHI Conference on Human Factors in Computing Systems*, pp. 987–996, ACM, 2012.

[57] J. Angulo and E. Wästlund, "Exploring touch-screen biometrics for user identification on smart phones," in *Privacy and Identity Management for Life*, pp. 130–143, Springer, 2012.

[58] 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 SIGCHI Conference on Human Factors in Computing Systems*, pp. 977–986, ACM, 2012.

[59] 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*, pp. 39–50, ACM, 2013.

[60] J. Sun, R. Zhang, J. Zhang, and Y. Zhang, "Touchin: Sightless two-factor authentication on multi-touch mobile devices," *arXiv preprint arXiv:1402.1216*, 2014.

[61] N. Sae-Bae and N. Memon, "Online signature verification on mobile devices," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 6, pp. 933–947, 2014.

[62] N. Sae-Bae and N. Memon, "A simple and effective method for online signature verification," in *Biometrics Special Interest Group (BIOSIG), 2013 International Conference of the*, pp. 1–12, IEEE, 2013.

[63] M. Martinez-Diaz, J. Fierrez, and J. Galbally, "The doodb graphical password database: data analysis and benchmark results," *IEEE Access*, vol. 1, pp. 596–605, 2013.

[64] M. Martinez-Diaz, J. Fierrez, J. Galbally, F. Alonso-Fernandez, and J. Ortega-Garcia, "Signature verification on handheld devices," in *Proc. MADRINET Workshop, Salamanca, Spain*, pp. 87–95, 2007.

[65] J. Koreman, A. Morris, D. Wu, S. Jassim, H. Sellahewa, J. Ehlers, G. Chollet, G. Aversano, H. Bredin, S. Garcia-Salicetti, *et al.*, "Multi-modal biometric authentication on the securephone pda," 2006.

[66] V. Iranmanesh, S. M. S. Ahmad, W. A. W. Adnan, S. Yussof, O. A. Arigbabu, and F. L. Malallah, "Online handwritten signature verification using neural network classifier based on principal component analysis," *The Scientific World Journal*, vol. 2014, 2014.

[67] S. M. S. Ahmad, A. Shakil, A. R. Ahmad, M. Agil, M. Balbed, and R. Anwar, "Sigma-a malaysian signaturesâĂŹ database," in *IEEE/ACS International Conference on Computer Systems and Applications,(AICCSA)*, pp. 919–920, IEEE, 2008.

[68] H. Xu, Y. Zhou, and M. R. Lyu, "Towards continuous and passive authentication via touch biometrics: An experimental study on smartphones," in *Symposium On Usable Privacy and Security (SOUPS 2014)*, USENIX Association, 2014.

[69] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee, "Individuality of handwriting," *Journal of Forensic Sciences*, vol. 47, no. 4, 2002.

[70] Ananda, "Signeasy announces new security features and enhancements for ios8 app, aims to streamline the way we access and sign digital paperwork." `http://blog.getsigneasy.com`, 2014.

[71] Sutisoft, "Signature verification." `http://www.sutisoft.com/sutidsignature/key-features.htm`, 2014.

[72] H. Khan, A. Atwater, and U. Hengartner, "Itus: an implicit authentication framework for android," in *proceedings of the 20th annual international conference on Mobile computing and networking*, pp. 507–518, ACM, 2014.

[73] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone.," in *USENIX annual technical conference*, vol. 14, Boston, MA, 2010.

[74] W. Lee, "Mobile apps and power consumption: Basics." `https://developer.qualcomm.com/blog/mobile-apps-and-power-consumption-basics-part-1`, 2013.

[75] W.-H. Lee and R. B. Lee, "Multi-sensor authentication to improve smartphone security," in *International Conference on Information Systems Security and Privacy*, 2015.

[76] C. Nickel, T. Wirtl, and C. Busch, "Authentication of smartphone users based on the way they walk using k-nn algorithm," in *8th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, pp. 16–20, IEEE, 2012.

[77] J. Sauro, "Measuring usability with the system usability scale (sus)." `http://www.measuringu.com/sus.php`, 2011.

[78] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008.

[79] A. Buriro, B. Crispo, and Y. Zhauniarovich, "Please hold on: Unobtrusive user authentication using smartphone's built-in sensors," in *IEEE International Conference on Identity, Security and Behavior Analysis (ISBA-2017)*, IEEE, 2017.

[80] O. Riva, C. Qin, K. Strauss, and D. Lymberopoulos, "Progressive Authentication: Deciding When to Authenticate on Mobile Phones," in *Proceedings of the 21st USENIX Conference on Security Symposium*, pp. 301–316, 2012.

[81] W. Shi, J. Yang, Y. Jiang, F. Yang, and Y. Xiong, "Senguard: Passive user identification on smartphones using multiple sensors," in *IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 141–148, 2011.

[82] N. Neverova *et al.*, "Learning human identity from motion patterns," *IEEE Access*, vol. 4, pp. 1810–1820, 2016.

[83] U. Mahbub *et al.*, "Active user authentication for smartphones: A challenge data set and benchmark results,"

[84] T. Vidas *et al.*, "All Your Droid Are Belong to Us: A Survey of Current Android Attacks," in *Proceedings of the 5th USENIX Conference on Offensive Technologies*, pp. 81–90, 2011.

[85] Y. Zhauniarovich *et al.*, "MOSES: Supporting and Enforcing Security Profiles on Smartphones," *IEEE Transactions on Dependable and Secure Computing*, vol. 11, pp. 211–223, May 2014.

[86] O. Riva *et al.*, "Progressive Authentication: Deciding When to Authenticate on Mobile Phones," in *Proceedings of the 21st USENIX Conference on Security Symposium*, pp. 301–316, 2012.

[87] E. Hayashi *et al.*, "CASA: Context-aware Scalable Authentication," in *Proceedings of the Ninth Symposium on Usable Privacy and Security*, pp. 3:1–3:10, 2013.

[88] H. Khan *et al.*, "Itus: An Implicit Authentication Framework for Android," in *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, pp. 507–518, 2014.

[89] M. Harbach *et al.*, "It's a Hard Lock Life: A Field Study of Smartphone (Un)Locking Behavior and Risk Perception," in *Proceedings of the Symposium On Usable Privacy and Security*, pp. 213–230, 2014.

[90] B. Priyantha, D. Lymberopoulos, and J. Liu, "Littlerock: Enabling energy-efficient continuous sensing on mobile phones," *IEEE Pervasive Computing*, vol. 10, no. 2, pp. 12–15, 2011.

[91] H. Lu *et al.*, "The jigsaw continuous sensing engine for mobile phone applications," in *Proceedings of the 8th ACM conference on embedded networked sensor systems*, pp. 71–84, ACM, 2010.

[92] M. E. Fathy, V. M. Patel, and R. Chellappa, "Face-based active authentication on mobile devices," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1687–1691, IEEE, 2015.

[93] P. Samangouei, V. M. Patel, and R. Chellappa, "Attribute-based continuous user authentication on mobile devices," in *Biometrics Theory, Applications and Systems (BTAS), 2015 IEEE 7th International Conference on*, pp. 1–8, IEEE, 2015.

[94] A. Roy, T. Halevi, and N. Memon, "An hmm-based behavior modeling approach for continuous mobile authentication," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3789–3793, IEEE, 2014.

[95] X. Zhao, T. Feng, and W. Shi, "Continuous mobile authentication using a novel graphic touch gesture feature," in *Biometrics: Theory, Applications and Systems (BTAS), 2013 IEEE Sixth International Conference on*, pp. 1–6, IEEE, 2013.

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

[97] M. O. Derawi, C. Nickel, P. Bours, and C. Busch, "Unobtrusive user-authentication on mobile phones using biometric gait recognition," in *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2010 Sixth International Conference on*, pp. 306–311, IEEE, 2010.

[98] E. Vildjiounaite, S.-M. Mäkelä, M. Lindholm, R. Riihimäki, V. Kyllönen, J. Mäntyjärvi, and H. Ailisto, "Unobtrusive multimodal biometrics for ensuring privacy and information security with personal devices," in *International Conference on Pervasive Computing*, pp. 187–201, Springer, 2006.

[99] H. Zhang, V. M. Patel, M. Fathy, and R. Chellappa, "Touch gesture-based active user authentication using dictionaries," in *2015 IEEE Winter Conference on Applications of Computer Vision*, pp. 207–214, IEEE, 2015.

[100] H. Saevanee, N. Clarke, S. Furnell, and V. Biscione, "Continuous user authentication using multi-modal biometrics," *Computers & Security*, vol. 53, pp. 234–246, 2015.

[101] S. Oishi, M. Ichino, and H. Yoshiura, "Fusion of iris and periocular user authentication by adaboost for mobile devices," in *2015 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 428–429, IEEE, 2015.

[102] R. Murmuria, A. Stavrou, D. Barbará, and D. Fleck, "Continuous authentication on mobile devices using power consumption, touch gestures and physical movement of users," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 405–424, Springer, 2015.

[103] V. M. Patel, R. Chellappa, D. Chandra, and B. Barbello, "Continuous user authentication on mobile devices: Recent progress and remaining challenges," *IEEE Signal Processing Magazine*, vol. 33, no. 4, pp. 49–61, 2016.

[104] C. Shen, T. Yu, S. Yuan, Y. Li, and X. Guan, "Performance analysis of motion-sensor behavior for user authentication on smartphones," *Sensors*, vol. 16, no. 3, p. 345, 2016.

[105] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang, "Silentsense: silent user identification via touch and movement behavioral biometrics," in *proceedings of the 19th annual international conference on Mobile computing & networking*, pp. 187–190, ACM, 2013.

[106] H. Gascon, S. Uellenbeck, C. Wolf, and K. Rieck, "Continuous authentication on mobile devices by analysis of typing motion behavior.," in *Sicherheit*, pp. 1–12, Citeseer, 2014.

[107] C. Shen, Y. Li, T. Yu, S. Yuan, X. Yi, and X. Guan, "Motion-sensor behavior analysis for continuous authentication on smartphones," in *Intelligent Control and Automation (WCICA), 2016 12th World Congress on*, pp. 2023–2028, IEEE, 2016.

[108] N. Z. Hamilton, "Correlation-based feature subset selection for machine learning," 1998.

[109] D. Tax, "Ddtools, the data description toolbox for matlab," June 2015. version 2.1.2.

[110] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, "How many trees in a random forest?," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pp. 154–168, Springer, 2012.

[111] P. S. Teh, P. S. Teh, N. Zhang, N. Zhang, A. B. J. Teoh, A. B. J. Teoh, K. Chen, and K. Chen, "Tdas: a touch dynamics based multi-factor authentication solution for mobile devices," *International Journal of Pervasive Computing and Communications*, vol. 12, no. 1, pp. 127–153, 2016.

[112] A. Ross and A. Jain, "Information fusion in biometrics," *Pattern recognition letters*, vol. 24, no. 13, pp. 2115–2125, 2003.

[113] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. Duin, "Is independence good for combining classifiers?," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 2, pp. 168–171, IEEE, 2000.

[114] A. Jain, P. Flynn, and A. A. Ross, *Handbook of biometrics*. Springer Science & Business Media, 2007.

[115] 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," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 136–148, 2013.

[116] Q. Tao and R. N. Veldhuis, "Biometric authentication for a mobile personal device," in *Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on*, pp. 1–3, IEEE, 2006.

[117] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 4, 2001.

[118] H. Saevanee, N. L. Clarke, and S. M. Furnell, "Multi-modal behavioural biometric authentication for mobile devices," in *IFIP International Information Security Conference*, pp. 465–474, Springer, 2012.

[119] H. Aronowitz, M. Li, O. Toledo-Ronen, S. Harary, A. Geva, S. Ben-David, A. Rendel, R. Hoory, N. Ratha, S. Pankanti, *et al.*, "Multi-modal biometrics for mobile authentication," in *Biometrics (IJCB), 2014 IEEE International Joint Conference on*, pp. 1–8, IEEE, 2014.

[120] M. A. Ferrer, A. Morales, C. M. Travieso, and J. B. Alonso, "Low cost multimodal biometric identification system based on hand geometry, palm and finger print texture," in *2007 41st Annual IEEE International Carnahan Conference on Security Technology*, pp. 52–58, IEEE, 2007.

[121] D.-S. Kim and K.-S. Hong, "Multimodal biometric authentication using teeth image and voice in mobile environment," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 4, pp. 1790–1797, 2008.

[122] C. McCool, S. Marcel, A. Hadid, M. Pietikäinen, P. Matejka, J. Cernockỳ, N. Poh, J. Kittler, A. Larcher, C. Levy, *et al.*, "Bi-modal person recognition on a mobile phone: using mobile phone data," in *Multimedia and Expo Workshops (ICMEW), 2012 IEEE International Conference on*, pp. 635–640, IEEE, 2012.

[123] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[124] B. Logan *et al.*, "Mel frequency cepstral coefficients for music modeling.," in *ISMIR*, 2000.

[125] C.-W. Hsu, C.-C. Chang, C.-J. Lin, *et al.*, "A practical guide to support vector classification," 2003.

Appendix A

Parameter Selection for Different Classifiers Used in the Chapter 7

Table A.1: Parameter Selection for BayesNET classifier on slide modality.

Naive bayes	Markov blanket	Max parents	Score type	Alpha	ms/model	EER
T	T	5	Entropy	0.25	64	0.1242
T	F	5	Entropy	0.25	90	0.1242
F	T	5	Entropy	0.25	114	0.1242
T	T	3	Entropy	0.75	60	0.1263
T	F	3	Entropy	0.75	92	0.1263
T	T	5	Entropy	0.50	64	0.1278
T	F	5	Entropy	0.50	83	0.1278
F	T	5	Entropy	0.50	106	0.1278
F	T	3	Entropy	0.75	93	0.1286
F	F	5	Entropy	0.25	83	0.1310
T	T	3	Entropy	0.25	56	0.1311
T	F	3	Entropy	0.25	83	0.1311
T	T	3	Entropy	0.50	58	0.1334
T	F	3	Entropy	0.50	80	0.1334
F	T	3	Entropy	0.25	85	0.1336
F	F	3	Entropy	0.25	52	0.1339
F	F	3	Entropy	0.50	72	0.1346
F	F	3	Entropy	0.75	59	0.1351
F	T	3	Entropy	0.50	82	0.1355
F	T	3	Bayes	0.25	119	0.1386

Table A.2: Parameter Selection for BayesNET classifier on pickup modality.

Naive bayes	Markov blanket	Max parents	Score type	Alpha	ms/model	EER
T	T	3	Bayes	0	762	0.2045
T	F	3	Bayes	0	898	0.2045
F	T	5	Bayes	0.5	6832	0.2102
F	F	3	Bayes	0.25	1283	0.2123
F	T	3	Bayes	0.25	1275	0.2137
T	T	5	Bayes	0.75	5642	0.2143
T	F	5	Bayes	0.75	5648	0.2143
T	T	3	BDeu	0	777	0.2155
T	F	3	BDeu	0	897	0.2155
T	F	5	Bayes	0.5	5645	0.2167
T	T	5	Bayes	0.5	5660	0.2167
F	F	5	Bayes	0.75	6880	0.2179
F	T	5	Bayes	0.75	6848	0.219
T	T	3	Bayes	0.25	885	0.2197
T	F	3	Bayes	0.25	905	0.2197
T	T	3	Bayes	0.5	883	0.22
T	F	3	Bayes	0.5	898	0.22
F	T	3	BDeu	0	1244	0.2211
T	F	3	Entropy	0	1795	0.2215
T	T	3	Entropy	0	2354	0.2215

Table A.3: Parameter Selection for BayesNET classifier on voice modality.

Naive bayes	Markov blanket	Max parents	Score type	Alpha	ms/model	EER
F	F	5	Entropy	0	205	0.2681
F	T	5	Entropy	0	306	0.2681
F	F	5	AIC	0.5	143	0.3001
F	F	0	AIC	0	162	0.3001
F	F	0	Bayes	0.5	163	0.3001
T	F	1	MDL	0.5	164	0.3001
F	F	1	Entropy	0.75	164	0.3001
F	F	0	BDeu	0.75	166	0.3001
F	T	0	BDeu	0.25	169	0.3001
T	F	0	Entropy	0.5	170	0.3001
T	F	1	Entropy	0.25	171	0.3001
F	T	0	BDeu	0.5	172	0.3001
F	F	0	Bayes	0.75	172	0.3001
T	F	0	MDL	0.25	173	0.3001
T	F	1	MDL	0.75	173	0.3001
T	F	0	Entropy	0.75	173	0.3001
F	F	0	Bayes	0.25	173	0.3001
T	T	1	AIC	0.25	175	0.3001
F	F	0	BDeu	0.5	175	0.3001
T	F	0	MDL	0.75	177	0.3001

Table A.4: Parameter Selection for SMO classifier on slide modality.

C	Gamma (γ)	Epsilon (ϵ)	Num folds	ms	EER
16	0.0625	1E-012	-1	8433	0.1864
0.25	0.0625	1E-012	-1	5865	0.1887
0.25	1	1E-012	-1	6018	0.1893
0.25	2	1E-012	-1	4717	0.1894
0.0625	2	1E-012	-1	4532	0.1899
4	1	1E-012	-1	3503	0.1903
1	2	1E-012	-1	3516	0.1903
1	1	1E-012	-1	4741	0.1908
0.0625	4	1E-012	-1	3598	0.1917
1	0.0625	1E-012	-1	4397	0.1946
0.25	4	1E-012	-1	3809	0.1952
16	1	1E-012	-1	3054	0.1979
4	2	1E-012	-1	3067	0.1984
0.0625	1	1E-012	-1	6501	0.1984
4	0.0625	1E-012	-1	7848	0.1984
16	4	1E-012	-1	2686	0.1993
16	2	1E-012	-1	2795	0.1993
16	0.00390625	1E-012	-1	4883	0.1996
4	4	1E-012	-1	2979	0.2002
1	4	1E-012	-1	3386	0.2002

Table A.5: Parameter Selection for Random Forest classifier on slide modality.

Num trees	ms/model	EER
1024	4453	0.1434
512	2255	0.1531
256	1167	0.1596
128	1012	0.177
64	655	0.1778
32	562	0.1884
16	443	0.238
8	380	0.2486
4	335	1
2	259	1

Table A.6: Parameter Selection for Random Forest classifier on pickup modality.

Num trees	ms/model	EER
256	13988	0.2083
512	31898	0.2083
128	13248	0.2331
64	5571	0.2601
32	6208	0.263
16	3055	0.2757
8	2689	0.2769
4	1671	0.2894
2	1286	1

Table A.7: Parameter Selection for Random Forest classifier on voice modality.

Num trees	ms/model	EER
512	4402	0.2452
256	2065	0.2485
1024	12251	0.2584
128	1209	0.2788
64	742	0.3554
32	563	0.5349
16	424	0.5736
8	352	0.5641
4	319	0.5516
2	251	0.5158

Table A.8: Feature Selection for the slide modality.

X	Y	Pressure	Size	Offset	EER
T	T	T	T	T	0.1178
T	T	T	T	F	0.1230
T	T	T	F	T	0.1363
T	T	F	T	T	0.1537
T	T	T	F	F	0.1542
F	T	T	T	T	0.1611
T	T	F	T	F	0.1629
F	T	T	T	F	0.1739
T	F	T	T	T	0.1809
T	F	T	T	F	0.1875
T	T	F	F	T	0.1944
T	F	T	F	T	0.2029
F	T	T	F	T	0.2062
T	T	F	F	F	0.2080
F	T	T	F	F	0.2171
F	T	F	T	T	0.2199
T	F	T	F	F	0.2287
F	T	F	T	F	0.2327
F	F	T	T	T	0.2523
T	F	F	T	T	0.2712

Table A.9: Feature Selection for the pickup/lift modality.

accX	accY	accZ	gyroX	gyroY	gyroZ	rotX	rotY	rotZ	time offset	EER
T	T	T	T	T	T	F	T	T	T	0.1589
T	T	F	F	T	T	F	T	T	T	0.1606
T	T	T	T	F	T	F	T	T	T	0.1633
T	T	F	F	F	T	F	T	T	T	0.1642
T	F	T	T	F	T	F	T	T	T	0.1648
F	T	T	F	F	T	F	T	T	T	0.1650
F	T	T	F	T	T	F	T	T	T	0.1655
T	F	T	T	T	T	F	T	T	T	0.1656
T	T	F	T	F	T	F	T	T	T	0.1656
T	T	T	F	T	T	F	T	T	T	0.1660
F	T	T	T	T	T	F	T	T	T	0.1661
T	T	T	F	F	T	F	T	T	T	0.1674
F	T	T	T	F	T	F	T	T	T	0.1681
T	F	F	F	T	T	F	T	T	T	0.1692
F	F	T	F	T	T	F	T	T	T	0.1697
T	T	F	T	T	T	F	T	F	T	0.1697
T	T	F	T	T	T	F	T	T	T	0.1699
T	T	T	F	T	T	F	T	T	F	0.1715
T	T	T	T	F	T	T	T	T	T	0.1717
T	F	T	F	F	T	F	T	T	T	0.1720

Appendix B

Demographic Questionnaire used in the Chapter 6

- Who you are?
  - Male
  - Female
  - I don't want to disclose
- How old you are?
  - $\leq$ than 20 years.
  - $>$ 20 years and $\leq$ 40 years.
  - $>$ 40 years and $\leq$ 60 years.
  - $>$ than 60 years.
  - I don't want to disclose
- Tell us about your nationality.
  - _____
  - I don't want to disclose
- Which hand(s) do you use for interacting with your smartphone?
  - Right
  - Left
  - Both
  - I don't want to disclose