Continuous Mobile Face Authentication

Christian Altenhofer BSc

MASTERARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Mobile Computing

in Hagenberg

im Dezember 2017

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Hagenberg, December 4, 2017

Christian Altenhofer BSc

Contents

Acknowledgements

I would like to thank all the people who contributed in some way to the work described in this thesis. First and foremost, I thank my master thesis advisor, Rainhard Findling MSc, for giving me the opportunity to be able to work and write on this thesis. Every result described in this thesis was accomplished with his help and support. I thank him for the systematic guidance and great effort he put into training me in this scientific field. Like a charm!

I also want to cordially thank my mother and father, Maria and Reinhard Altenhofer, and my brother Martin Altenhofer, for the continuous support and help specially during hard times.

Further, I want to thank Dr. Christoph Schaffer and the whole Mobile Computing staff who made this thesis and my academic career possible and for all the opportunities provided during the studies of Mobile Computing.

Finally I want to thank all of my friends my aunts and uncles for their support and being test dummies for my master thesis application.

Abstract

There are plenty of other authentication approaches on the market or in research but most of them are active. Active means that users have to explicitly do something in order to be authenticated by the system which demands the users attention and time. In this work we present our continuous mobile face authentication approach to distinguish between the owner of a smartphone and possible attackers, just by using video sequences or camera streams captured by the front facing camera. Our approach does not demand explicit user actions. The authentication is continuous, which means that the system does not require users to enter credentials at a certain point. Continuous authentication is another way to prevent unauthorized access to the mobile device and works passively in the background of the smartphone. In this work we explain our continuous mobile face authentication approach with our three design goals: unobtrusive, continuous and mobile. The authentication system should have a reasonable authentication performance. We develop and evaluate our approach in three steps in order to evaluate weaknesses and do improvements in the next step. Our first implementation is a prototypical implementation of a face recognition system which is able to recognize faces in images and distinguish between different people. This prototype however is not able to detect people beyond the test database which is the major improvement in the second step of the implementation process. Also a very important part in the second step is the weight function module which weights the importance of the face observation sample depending on the elapsed time between two observations. Another part of this implementation step is the recording of the face-database as source for evaluating our approach. In the third and last step a time decay module is introduced. This is necessary to give the confidence value a decay over time when for a period no new face samples can be detected to avoid an allegedly attack. Concluding, our results indicate that our continuous mobile face authentication is a viable approach for face authentication with a good performance specially when detecting allegedly attackers.

Chapter 1

Introduction

Nowadays nearly every person carries a smartphone and the amount of mobile devices is increasing. Many people store large amounts of data onto those devices to access the data anywhere and anytime. Some of this data is considered to be private and therefore needs protection. Mobile device security and privacy has become a popular issue nowadays, because mobile devices are prone to be lost or potentially being stolen due to their small sizes. If a smartphone without protection gets into the wrong hands, private data as well as installed applications can easily be misuse for sending emails or SMS, or more crucial making unwanted cash transactions with financial applications.

PIN codes and graphical pattern are a very frequently used technique to gain access to a locked system. Most people tend to prefer a drawn pattern over a sequence of digits though. Those authentication techniques are easy to use but have some serious downsides as well. Oily residues or so called ‘smudges‘ on the display surface are one side effect of touch patterns [3]. Next to the security issues conventional authentication methods come up with, they also effect the smartphone user experience negatively. A field study by Harbach et al. [30] found out that the average out of 260 participants spent around 2.9% of their smartphone interaction time just on authenticating. The participants who used a PIN or an unlock pattern considered it as unnecessary in 24.1% of the time which can lead to the situation where users fully renounces the security mechanism and make the system insecure and accessible for everyone. Another frequently used technique in nowadays smartphones is fingerprint authentication with the built-in fingerprint sensor which gives an easy to use and secure access to your device. With all the positive sides the fingerprint authentication comes with, also some negative side effects come along.

There is already a vast amount of authentication approaches on the market or in research but most of them are explicit, such as fingerprint authentication. Explicit means that users have to explicitly do something, such as touching the fingerprint sensor, to be authenticated on the system. This demands user attention and time, so the user experience is negatively affected by those points. Our approach is able to distinguish between the owner of a smartphone and other users (possible attackers) just by using video sequences or camera streams captured by the front facing camera.

The main motivation for our continuous mobile face authentication approach is to create an unobtrusive, continuous and mobile face authentication prototype. This means the system should not interrupt the user in any way, like force him to enter a PIN or a password at any point. Users just have to work on the device, and automatically remain authenticated when they have the permission to.

We review the currently most widely used and most important mobile state of the art and biometrical authentication approaches in chapter 2. In chapter 3, we provide an overview of the related work with focus on continuous authentication as well as face authentication approaches. We describe building blocks which are required for our authentication approach in chapter 4. In chapter 5 we explain our approach in detail with data acquisition and preprocessing, the difference image creation, the face recognition, weight and time decay function modules and the finally how to come up with a confidence score calculation. We present the recorded face-database as source for evaluating our approach in chapter 6. In chapter 7 we present the evaluation results of our continuous mobile face authentication approach. Finally, we conclude and provide an outlook in chapter 8.

Chapter 2

Mobile Authentication

In this chapter we survey some well known state of the art authentication techniques for mobile devices and discuss their advantages and disadvantages. We have a closer look on authentication and its core features especially with mobile devices. If users want to enter a secure room or want to gain access to a locked system they need to verify themselves. Users need to perform authentication so that the system knows they are allowed to enter the room or unlock the device. Burr et al. [8] describe authentication as the procedure of validating the identity of users or information systems. When authentication used in security context, it needs some kind of secret. A secret is something the user or system possesses and controls that is used to authenticate the identity of the claimant. Burr split up authentication into three types.

1. **Knowledge-based:** *something users know* Persons achieve authentication by testing their knowledge of something secret against information obtained from the system. Like a password, an answer to a security question, or an ID number.
2. **Possession object-based:** *something you have.* Authentication requires something in the physical sense. The claimant has to proof his identity by possessing some physical token. The prevalent examples for this strategy are security token, ID cards or other trusted devices.
3. **Biometric:** *something you are.* The authentication is based on physical or behavioral characteristics of the user. This can be represented by one or more attributes.

Mobile authentication mostly relies on knowledge-based strategies like passwords or PINs. In some cases also object-based strategies are used like using bluetooth beacon or other devices for verification. Biometrics are less frequently used in mobile authentication with the exception of the increasing usage of built-in fingerprint sensors in smartphones (see section 2.2.1).

2.1 State of the Art Mobile Authentication

With the increasing amount of smartphones, mobile device security and privacy has become a popular issue nowadays. Mobile devices are prone to be lost or potentially being stolen due to their small sizes. If the attacker has access to the smartphone he has access to private data or he can easily misuse the installed applications like online social networks as well as more crucial financial applications [2]. According to the online report of Donna Tapellini [67], the total number of stolen devices in the USA increased from 1.6 million in 2012 to 3.1 million in 2013, and rising. A user study of 2016 by Harbach et al. [31] shows that 68% of a total amount of 8000 participants protect their smartphone sufficient by PIN or security pattern. In the other 32% the data is freely available in case the phone is stolen or lost. Those 8000 participants could chose in the studies questionnaire how sensitive they would rate the data on the smartphone on a scale from 1 to 7. Those without any security mechanism scored an average of 3.44 and those with PIN or security pattern scored 4.54. The reason why most participants don't secure their smartphone sufficiently is, that it is inconvenient to enter those secrets and that they want to access the device functionality faster and do not want to enter a secret before hand to access the phone. Some other participants stated that it is hard for them to memorize the PIN or security pattern or that they didn't even think about security for handhelds [7].

2.1.1 PIN

PIN codes are a very frequently used technique to gain access to a locked system. For example when drawing money from an ATM the PIN is used as second level of security additional to the credit card. Nevertheless PINs often are the only level of security like on smartphone lock screens. Users gain access to personal data just by entering a 4 digit number. When using conventional PINs the user can choose digits between 0 and 9 which give a total amount of 10000 possibilities (10^4) which could be brute forced.

2.1.2 Graphical Pattern

Some people tend to prefer a drawn pattern over a sequence of digits. In 2008 Android introduced its optional pattern lock system, which requires a finger swipe over a particular sequence of dots to unlock the smartphone (see in figure 2.1). Users can decide between different amounts of connectible dots. Although being more error prone and slower than conventional PINs, users prefer to draw a pattern on the screen due its ease and playful usage [15].

2.1.3 Attacks

PIN and graphical pattern authentication techniques are easy to use but have some serious downsides. Oily residues or so called ‘smudges‘ (see in figure 2.1) on the display surface are one side effect of touch patterns which are frequently used to unlock the smartphone. Aviv et al. [3] states that those smudge attacks are a threat for three reasons:

- The draw residual is surprisingly long visible.
- Smudges are hard to delete through wiping or pocketing the device.
- The oily residual of the pattern is easy to collect and analyze.

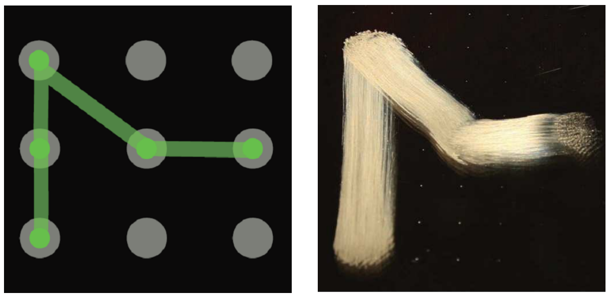

(a) (b)

Figure 2.1: (a) A drawn sequence of dots. (b) the oily residual ‘smudge‘ as result [15].

The three reasons which Aviv et al. state are demonstrated in their experiments. They unlocked devices by exposing the graphical pattern only with the oily leftovers on the device screen. These leftovers can be easily captured by a camera. The pattern can be even more visible for the human eye by using a commercial photo editing package for lighting and color adjustments. So if a smartphone will be stolen with authentication mechanism like this, it is pretty likely that the attacker is able to access the private data or even passwords for social media or banking just by using a commercial camera.

Another security risk when using PIN or unlock pattern is a so called ‘shoulder-surfing‘ attack. The attacker can obtain the secret by direct observation or by recording how users authenticate. This is a known risk of special concern when authenticating in public places [42].

2.1.4 Usability

Next to the security issues conventional authentication methods come up with, they also effect the smartphone user experience negatively. A field study by Harbach et al. [30] found out the average out of 260 participants spent around 2.9% of their smartphone interaction time just on authenticating. The participants who used a PIN or an unlock pattern considered it as unnecessary in 24.1% of the time which can lead to the situation where users fully renounces the security mechanism and make the system insecure. Users do not want to spend time entering a password or PIN or security pattern to access their smartphone [69]. Nevertheless users tend to use PINs, passwords or unlock pattern. The reason therefor is that those are the only authentication methods they are aware of unfortunately. Most of the PINs are chosen pretty weak. This is because users have to remember many different passwords and PINs. They choose easily guessable passwords and reuse those out of self-defense, because this is the only way to remember a vast amount secrets which are needed for the huge amount of different accounts they use [13]. Most authentication systems are evaluated against two properties: security and usability [74]. The system often has to make a tradeoff between those two properties. Users will either choose a easy to remember password which is insecure or they will choose a complex password which they cannot remember properly and have to write it down. Hence, the top reason why users disable their security mechanism to enter the phone is inconvenience and the negative effect on usability [20, 30, 31].

2.2 Biometric Authentication

Beside the already mentioned state of the art authentication techniques for mobile devices (see section 2.1) biometrics can be used for verification as well. Biometric authentication can be split in two categories: behavioral and physiological. Physiological authentication techniques often requires additional sensors like a fingerprint sensor the to acquire the wanted biometric. Others can use e.g. the camera sensor which can be very battery draining when used inefficient. The usage of behavioral biometrics however can be obtained using existing parts such as the smartphone keyboard or display, when using typing pattern for example. The most outstanding advantage of behavioral biometrics is that they do not afford the users attention and time. That is why behavioral techniques make the authentication process attractive to users due to their unobtrusive usage [5]. Biometric recognition algorithms improve more and more nowadays with the sensors newer smartphones come up with by default and the improvement of processing power it is easy to use physiological biometrics like the face or the voice as authentication feature [69]. Figure 2.2 from [2] gives an overview about the most common biometric authentication techniques. Some more detailed explanations of some state of the art biometric authentication techniques are stated in chapter 3. To encounter weakly chosen passwords biometrical authentication methods might come pretty handy. A study by De Luca and Lindqvist [15] states that an average smartphone user unlocks the phone about 50 times a day. Another user study by Hintze et al. [33] came up with a total of amount of 60 smartphone interactions average per day where in

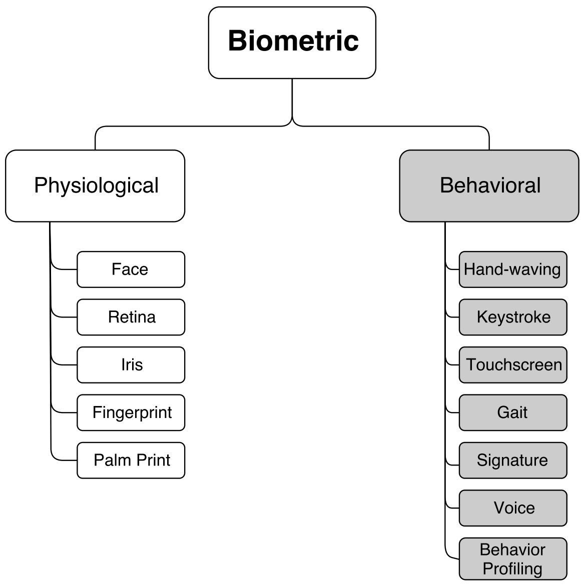


Figure 2.2: Biometric approaches to authenticate users (adopted from [2]).

45% (27 interactions) the phone was unlocked with an average session length of 307 seconds (Other interactions are not security sensitive e.g. switching to another song on media player or checking the time). That means that the biggest challenge beside security is creating and designing a security mechanism which is easy and fast to use or most of the users will disable it. A metric which is used the measure the time a user needs to provide a sample for authentication is called ‘user action time‘. This metric neither includes processing time spent verifying the quality of the sample nor the authentication and server response time. It only measures the time the users need for e.g. taking photos for face authentication. The user study of Trevin et al. [69] states that voice is the fastest with a median user action time of 5.15 seconds, followed by face authentication with 5.55 seconds. Password has the highest user action time with 7.46 seconds when considering only those three authentication methods. Hence, biometric approaches give a convenient and unobtrusive user experience which passwords and PINs are not able to give. They give the user the opportunity to enter the system fast and in an easy way.

"‘[...]A key selling point of biometric authentication is that it allows users to move away from passwords, both for use in authenticating to third parties, and for unlocking their own physical device. This eliminates the requirement to enter passwords, and avoids the inconvenience of forgetting passwords.[...]"’ (Greig Paul 2016, IEDs on the Road to Fingerprint Authentication : Biomet-

rics have vulnerabilities that PINs and passwords don't) [56]

2.2.1 Fingerprint Authentication

Fingerprint authentication refers to the automated method of verifying users by their individual fingerprints. Mobile fingerprint authentication uses a built-in fingerprint sensor to access the smartphone which gives an easy to use and more secure access to your device. A user study on smartphone security and usage by Breitinger et al. [7] asked 548 participants which authentication mechanism they would prefer. Out of those which are willing to use biometrical authentication techniques, about 87% voted for fingerprint. A usability study of Karthikeyan et al. [40] gave 40 participants tasks to find out if they would prefer fingerprint or PIN as smartphone unlocking method. Those tasks included setting up the technique, unlock the smartphone, download an application from the store and change the pattern / the fingerprint. The analysis shows that fingerprint takes much more time for setting up. The average setup time of the PIN was about 24 seconds. Fingerprint setup phase in comparison took an average participant about 46 seconds. The attendees however preferred the fingerprint technique. Only 20% out of 40 participants would like to use a PIN over a single tap on the fingerprint-sensor to unlock the smartphone. Another 30% do not mind which technique they use and half of the tested people like the fingerprint method more.

2.2.2 Voice Authentication

Voice authentication is another biometric method where users are identified based on the way and pattern of speaking [2]. The most outstanding advantage when using voice features as authentication method is that the authentication process can be done remotely [36]. The claimant is able to get access to a locked room or system via speaking over the telephone system. The crucial part when using this kind of authentication is that other people are able to hear your spoken voice and note down for example a spoken PIN or any other personal data. To protect voice authentication Ji et al. [36] created a system architecture which allows the claimant to whisper the authentication term into the device microphone. To improve the speech recognition performance a high-pass filter and an acoustic model adaption algorithm was added to the voice. To add another layer of security Ji et al. introduced a one time password (OTP) which is composed of several digits. The user is only verified as authorized candidate when the system recognizes his voice as trusted voice and when he speaks the OTP correctly. With additional improving steps like adding a high-pass filter, the experimental results showed that voice authentication can replace current identification methods such as passwords and PINs. The additional security level (OTP) makes this approach more secure because to the two-factor authentication.

2.2.3 Gait Authentication

Gait recognition is a biometric method where users get recognized by the way they walk [17]. This is a continuous approach to verify a user in an unobtrusive way. The three main approaches in biometric gait recognition are: machine vision based, floor sensor based and wearable sensor based gait recognition. The machine vision based version of gait recognition uses computer vision and basically needs several cameras. This cameras track suitable optics on the person to acquire the gait data [45, 51]. Some techniques like background segmentation are used to extract the features of the tracked person. With this approach non-coorperative persons can be detected at a distance in real-world changing environmental conditions.

In the floor sensor based approach the sensors are placed on mats or on other things in the floor. The pressure-measuring mat obtain the footprints. Based on the direction, the position and the distance of the footprints to each other the system is able to distinguish between different persons [48].

Wearable sensor based gait recognition approach is based on wearing motion recording sensors on different body parts for example a smartphone with the required sensors. The different features can be acceleration (measured by accelerometers), rotation and number of degrees per second of rotation (measured by gyroscope sensors), force applied when walking (measured by force sensors) and some others [17]. Due to the fact that most of the given sensors are already built-in in smartphones means that modern smartphones are a well suited base for an unobtrusive continuous authentication approach based on biometric gait [17, 34].

2.2.4 Face Authentication

Another biometrics for mobile authentication would be face recognition (see section 4.2). The user is able to enter the room or unlock a system using his face as ‘key‘. The verification of the face is done by images recorded by e.g. the built-in camera of a smartphone [18, 66]. The face authentication process is split up in three main steps [55]. The first step is to acquire faces of the claimant out of images or video sequences. In the second step features of the face gets extracted. Finally these extracted features are passed on to the classifier to distinguish between different users. An additional step before the feature extraction which is not explicitly stated in the paper of Patel et al. [55] is preprocessing of the image. This could consists of the following steps: Detecting the face in the image, crop the face part, make the image gray scale, apply histogram equalization for better contrast and resize the image to reduce the amount of features [1, 4, 21]. However some challenges have to be overcome before successfully using faces as authentication biometric. Faces may have very poor illumination, different pose variations or missing facial parts which the system has to deal with [55], which maybe lead to unsuccessful authentication attempts.

2.2.5 Drawbacks of Biometric Authentication

Beside the easy and convenient usage of biometric authentication, the biometrics do have some security issues as well. The permanent and non-revocable characteristic of a biometric, such as fingerprint means that once it is captured and stored it will be lifelong valid. We leave fingerprints behind on almost every surface we touch [56]. This leaves the concern that people can not even prevent the fact that some strangers might have their identity stolen with a simple glass bottle they thrown away a few minutes ago. Or they have most likely an image of our face just by goggling our name. The trade-off between convenient usage and security has to be considered when biometrics are used as authentication mechanism. A good extension could be that applications with high security needs would get an additional security technique. This is called a two-factor authentication which could be an additional secret (PIN, password or any other secret) which has to be entered before accessing the application [59]. This could encounter the stated security issue of fingerprints but let the user enjoy the easy access to the smartphone. Only when using applications with high security needs the two-factor authentication comes in place which might introduce some usability issue and affect the positive user experience.

Another downside of biometric methods is that errors can appear which are not able to happen when we use PINs or passwords. The main two errors which may occur are Failure to Enroll (FTE) and Failure to Acquire (FTA) [69]. An Failure to Enroll error occurs when the participant is not able to use the biometric system. An example when using face authentication could be that the face verification engine is not able detect a face in the taken image. Failure to Acquire (FTA) errors are stated as follows. The participant is not able to provide a sample of sufficient quality. When using face authentication a given threshold of certainty needs to be reached to meet a predefined quality criteria.

A big concern when using biometrics for authentication is a so called spoof attack where the attacker imitates the users biometric. Spoof attacks like on faces authentication systems can be launched rather easy via printed photos, video replays or 3D masks [53]. Faces are easier to obtain in contrast to other biometrics like fingerprint or gait. A video or an image found online is enough for an attacker to commit a spoof attack on the face authentication system [54]. Figure 2.3 from [54] shows an attempt where an attacker uses a online video resource to unlock the smartphone. The biggest concern when using biometrics is that users can not change who they are. It is impossible to change your face, your iris pattern or your fingerprint. Which means

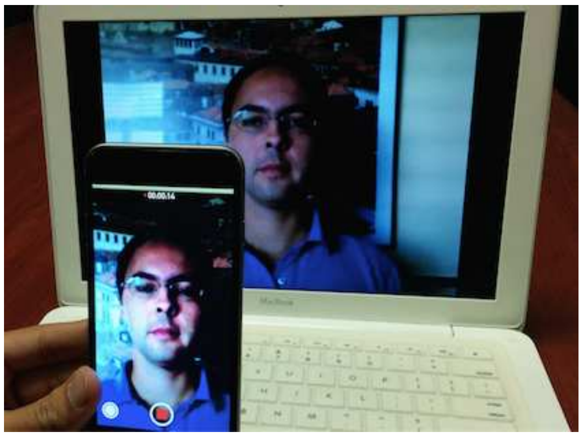


Figure 2.3: Demonstration of a video replay attack where an image sequence is collected via smartphone from a notebook screen (the spoof medium) from a video found online [54].

if someone is able to duplicate, steal or replace the users biometric it is possible for the attacker to authenticate to a biometric authentication system pretending to be someone else [11].

2.2.6 Summary

The most outstanding advantage of biometrics is that most can be used in an unobtrusive way in comparison to PIN and passwords, which always requires the users attention. Accessing a system or unlocking the smartphone does not require users to remember a secret which needs to be entered to verify the identity. A simple tap on the fingerprint sensor or a look in the camera is enough to access the system. Nevertheless, some concerns are given. When attackers are able to obtain some of the users biometric it is possible for them to access the biometric authentication system. The biggest concern however is that most biometrics are static and do not change over time. Which means users cannot change who they are, which is a problem once the biometric is obtained by an attacker. When users are able to decide if they use either biometrics or passwords as authentication technique the tradeoff between security and convenience has to be considered. Users have to choose what is more important to them and decide based on this.

Chapter 3

Related Work

This chapter gives an overview of state of the art research that is related to our continuous mobile authentication approach. We discuss similarities, advantages, disadvantages and core features of the given related work and describe how those approaches work.

3.1 Active Versus Passive Authentication

Authentication can be either active or passive which means if the user has to make active steps to verify himself or not [2].
Active authentication requires the user to enter some valid information to the device. Active authentication, like entering a password or PIN number, is mostly used as so called 'entry-point authentication', which requires the users attention and dedicated user time. Entry-point authentication means that users have to enter a secret correctly to pass the entry-point to the device, which is a certain drawback in usability. A drastic shortcoming of those entry-point authentication systems might be that the claimant only have to authenticate himself at one certain time. Once the allegedly attacker passed this point, he is able to fully access the smartphone, and without any additional security mechanisms he is possibly able to change the password of the entry-point. If this happens the user is not able to enter the enter-point by himself and is locked out of his own system.
Passive authentication, also known as implicit, progressive or continuous authentication, does not require users to enter credentials at a certain point. Continuous authentication offers another way to prevent unauthorized access to the smartphone and works passively and nonintrusive in the background of the smartphone [24].

In most cases passive authentication is divided into two phases [2]. The enrollment phase is a learning process. Users are working on the smartphone as usual, the system though records suitable features for a period of time. If for example touchscreen analysis is used, the recorded features may include touch speed, movement or X/Y-coordinates of the touches. The system learns characteristics of the user behavior by performing statistical analysis or using machine learning on the recorded data and creates a user model. The second, continuous authentication phase takes into account after the creation of the user model (enrollment). After the successful active login of the user, the system compares the current behavior with the learned user model, to recognize the user or a possible threat.

A continuous authentication approach should be light-weight, nonintrusive and as the name already indicates, continuous [2]. This means the system should not interrupt users in any way, like force them to enter a PIN or a password at any point, and the approach should use low computational power to save battery especially on mobile devices.

3.2 Continuous Biometric Authentication for Mobile and Desktop

Before discussing continuous face authentication we highlight other biometrics that can be used continuously or have other relevance to our approach.

When we are looking at continuous authentication techniques most of the time behavioral biometrics are used. Continuous techniques have the advantage that the verification of the user is done over and over. In entry-point authentication systems the claimant only has to verify himself once. Once access is granted no further verification is needed to use the system and get access to sensitive data. This could happen when users leave the workstation for a short break to get for example some documents from the printer. Anyone can use the computer in the meantime and pretend to be the authorized user. The following sections will give examples of biometrics which are good examples for biometrics frequently used in a continuous manner. The highlighted biometrics are keystroke recognition, touch behavior recognition and gait recognition. It will be discussed how those approaches acquire data, preprocess data and use this data for user classification. Figure 3.1 gives an overview of the general modules used in a biometric authentication system.

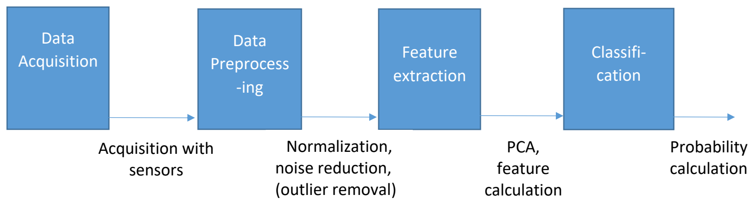


Figure 3.1: General overview of a biometric authentication system. Specific biometrics can have additional modules.

3.2.1 Keystroke Pattern Authentication

Keystroke biometric systems are identifying people by their typing style, which is unique in the hold time of keys or the time between keystrokes. Those typing characteristics are believed to be unique for different users and hard to duplicate [6]. In the following section classifiers like Support Vector Machines (SVM) and k-nearest-neighbors (KNN) are mentioned which will be explained in detail later on in section 4.2.1 and section 4.2.2.

Keystroke recognition is an inexpensive method for user authentication. The only requirements are a desktop computer or a notebook and a keyboard. Most approaches measure the times between typed keys or times between a group of typed keys. The unobtrusive technique is very handy for users because this get rid of the active verification process. One consideration is low volume computer input (drinking coffee, only using one hand for a short time window), which will need a predefined threshold. This threshold defines the quantity of data required for reasonable authentication if the input is below the threshold the user access is no longer granted [47]. Some approaches have defined some special cases when the system deals with long text inputs [68]. There are infrequently used keys which need reasonable values and computability for all feature measurements as well as the frequently used ones. When the typing frequency of a specific key is below a given threshold the mean value is calculated of the weighted average sum of the key in question and the average sum of an appropriate fallback group of keys (which is defined before hand). Other approaches use a sequence of typed keys so called n-graphs and the duration when a key is depressed [28, 78]. This is used to calculate the elapsed time between the depression of the first and last key of a sequence which is called 'duration of a n-graph' (this is the combination of the latencies between keystrokes and their durations). All this times and latencies can be used in different ways as features for further recognition purpose.

In some keystroke recognition systems it is necessary to remove outliers because users can have a long pause between key presses, for multiple reasons, which lead to long transition times that skew the feature measurement [47, 68]. Therefor every duration or transition time which is longer than a predefined threshold will be removed. Values are standardized into a specific range (e.g. between 0-1 by clamping the measurements at plus and minus two standard deviations of all samples from all participants). This standardizing is used to give each sample roughly the same weight. Other approaches will use only sequence of typed keys which are common in two or more typing samples, uncommon key sequences will be dropped out [28].

To extract features from keystroke data some approaches calculate the mean and standard deviations of key press durations and digraph transition times (e.g. between letters and non-letters or groups thereof) [47, 68]. Additional features could be the percentage of special keys usage. This percentage features are designed to capture the user's preference of certain keys or key groups. A typing sample can also be represented by its n-graphs [28] (sequence of typed keys) together with the duration of each n-graph. The typing samples can be represented in digraphs, trigraphs and so on which will be used as features than. The digraphs for the word 'authentication' could be for example: $(au, th, en, ti, ca, ti, on)$ and the trigraphs: (aut, hen, tic, ati). If the typed text is sufficiently long the same n-graph may occur more than once. In such cases the n-graph is reported only once to the system and the mean duration of its occurrences is used for later on classification.

For authenticating keystroke pattern the biometric needs to be transformed into a feature vector [47, 68]. Some of this feature vectors will also be generated earlier in the enrollment phase (template feature vectors). In the enrollment phase this template feature vectors will be generated and labeled. To get a two-class problem $\{user, threat\}$ out of a multi-class problem $\{person1, person2, person3, ...\}$ the feature vector gets transformed into feature-difference space by calculating vector distances between pairs of samples. Vector distances between sample pairs of the same person (within-person / verified user) and vector distances between pairs of samples of different persons (between-person / threat) are calculated, labeled and saved in a training dataset. A similar approach is done in our continuous mobile face authentication system (see section 5.3). In the authentication process a user's keystroke sample is converted again into a feature vector. Later on the difference vector of this feature vector and earlier obtained template feature vector gets calculated and used for authentication. Classifiers like KNN can be used to classify the difference feature vector of the current sample by comparing it with those in the training dataset. The observed difference feature vector gets assigned to one of the two classes $\{user, threat\}$ corresponding to the template difference vector with the smallest distance to.

Some approaches decide whether the current sample is from a verified user or not each time a new sample gets observed. The sample will be classified and a value will be calculated. This value (confidence score) which is in a nutshell, the probability of how certain the current user is a verified user, will be used to accept or reject the current user and bring the system into attack-state when rejected [78]. The new confidence score gets calculated by the current observation and the previous confidence score, both will be weighted with a specific weight function. A similar approach is done in our continuous mobile face authentication system (see section 5.5). If the confidence score drops below a certain predefined threshold and the system goes into the attack-state additional steps to verify the identity may be done. Other key stroke authentication systems use some different approaches to deal with the continuous aspect [47, 68]. Keystroke authentication systems have to deal with possible interruptions where the users leave the desk and do not provide input. A fixed time window can be used which is used as re-authentication step after a pause where the system collects and analyzes the typing pattern and decide based on this analysis if the user is verified or a threat for the system. The length of this time window is depending on a tradeoff between accuracy and reaction time. If the value at the end of the time window is below a certain threshold the user will be logged out. If the window length is chosen long the accuracy will increase but the system need longer time to react. If the window length is chosen short the accuracy will decrease but the system will react faster which influences the usability positive.

3.2.2 Touch Analytics Authentication

Touch analytics on smartphones is nearly the same as keystroke detection on Desktops. In touch analytics the swipe and type pattern on the display are measured and users get later on identified by the way they interact with the smartphone touchscreen. In the following section classifiers like SVM and KNN are mentioned which will be explained in detail later on in section 4.2.1 and section 4.2.2.

To gather the needed touch gestures the users raw touch data needs to be recorded. For each present finger the x and y coordinates as well as the pressure on the screen and the area of screen covered by the finger is recorded and could be used for generating features later on [24, 43]. This raw touch data is in some cases provided by the standard system API (e.g. Android system) [24] or from collection software which obtains data from the operating system [22]. Other mobile approaches use the type pattern on a software keyboard of the smartphone to gather touch data plus the smartphone orientation, torque of the smartphone, relative position of the smartphone and the timestamps of key press events. The additional data gets provided by the sensors in the smartphone like accelerometer, gyroscope and the software keyboard [25]. To ensure that only motion data related to the typing behavior is recorded time constraints need to be considered when signals are preprocessed [25]. The goal is to calculate a data point of all values during a certain time period T once the user has started typing. Every T seconds a data point gets calculated if the user does not introduce some typing data during a time period T_{stop} the system assumes that the user stopped the typing task. When T_1 is followed by a T_{stop} which means that the user stopped typing for the predefined period T_{stop} all sensor data after T_1 can be discarded. This reduces the unnecessary data and decrease the computation effort. Additional information like the user's dominant hand or if users use both hands or only one hand can be used as feature as well. This information can also be obtained by the touch coordinates on the screen [61].

An important step in touch analytic approaches is data normalization and standardization to generate touch feature vectors with the same size [24, 25]. This is important because the speed of the swipe gesture or the taps may vary. In addition artifacts or constant values such as the gravitational force on earth $9.81 m/s^2$ can be removed to reduce unnecessary data [24].

Some approaches extract up to 30 behavioral touch features that can be extracted from raw touchscreen data (e.g., finger up, finger down, finger move and multitouch) and additional features like event timings and device orientation [24, 77]. The first step of feature-extraction is to split up the touch observation data into individual strokes. A stroke is a sequence of touch data that begins with touching the screen and ends with lifting the finger. One stroke s is encoded as a sequence of vectors and can hold following data: start and stop location of the stroke, time stamp, the pressure on the screen, the area occluded by the finger, the orientation of the finger and the orientation of the phone [24]. Between two strokes s^m and s^{m+1} no input is recorded on the touchscreen. Examples for extracted features are e.g. ‘mid-stroke area covered‘, ‘average direction‘ or ‘stroke duration‘. Other approaches uses single taps and the corresponding location and duration as features [22]. All this features are later on used for classifying users.

The classifiers work on individual single strokes. For a more robust classification result multiple strokes can be combined together in an earlier stage. The number of strokes used for classification is very important and introduces a trade off between robustness of the classification and the time needed to detect a threat. During the authentication phase, the system continuously tracks all strokes and the classifier estimates if they were made by the legitimate user or by an attacker. In some approaches a predefined amount of consecutive negative classification results (classifier estimate an attacker) has to be reached to bring the system into attack state and back to a initial entry-point based authentication method [24, 43]. When multiple consecutive negative classifications are allowed some kind of counter is needed to keep track of the amount of negative classifications. Approaches which rely on taps rather than on strokes need some kind of sliding window. The sliding window defines the amount of taps needed for authentication. Only using single taps would not work in this kind of approach because of the huge similarity between taps of different users. This sliding window contains a predefined amount of ‘input units‘ (a combination of multiple taps). If none of the input units within the sliding window is from an authorized user the system will lock out the user [22]. The size of the sliding window is again a trade off between robustness of the classification and the time needed to detect a threat.

3.2.3 Gait Authentication

In the following section classifiers like SVM, models like hidden Markov model (HMM) as well as preprocessing techniques like principal component analysis (PCA) are mentioned which will be explained in detail later on in section 4.2.1, section 4.3 and section 4.4. Another biometric which can be used in the continuous authentication domain is gait recognition. Simply attaching a smartphone to a users pocket is enough to collect the time discrete combination of acceleration values of the built-in three dimensional accelerometer sensor during walking [17, 34]. The gait recognition approach can be adapted to a desktop version by analyzing the gait pattern with computer vision using a sequence of images captured with cameras [37, 45, 51]. The assumption here is that walking is a cyclic motion which repeats at a stable frequency. Poses in human walking cycles are the same for each person e.g. the arms and legs move forward and backward in a similar way among normal people. Differences exists in the phase of poses during the walking cycle e.g. the extend of arms or the shape of the torso which can become features for gait authentication [45]. Classification can be done on those features later on.

Visual gait recognition approaches where persons' contours are used for recognition rely on normalized sizes and alignment. Therefor the extracted images are size normalized where the silhouette has to fit to a fixed height and need to be centered in the image [45]. Also some filters can be applied to reduce noise. Some systems use an erosion filter to reduce noise and apply a weighted low pass filter in addition to trace the left and right body contours since only the contour of the person is needed here [37]. A gait energy image (GEI) which is used in [45] represents a human motion cycle in a single image which leads to the limitation of training templates since every person rely on only a few or just one GEI. To overcome those limitations a series of new GEI can be generated by analyzing the human silhouette distortion (different shoes, different clothes or different floor surfaces can affect the silhouette). The new GEI templates are generated by determine the range of distortion area and use another part of the template to fit into the distortion area. By repeating this step a new series of template GEIs is generated. PCA can be used for extracting features for recognition. Where most computer vision based gait recognition approaches on desktop computers can provide proper time-stamps to their observations, the raw gait signals which are acquired by low-quality accelerometers of the smartphone will struggle here, due to the inaccurate sampling rates of the mobile sensors and the huge occurrence of noise. To encounter the sampling rate problem some approaches use linear interpolation between the samples to ensure correct values at specific observation times. In addition multi-level wavelet decomposition to set explicit coefficients to 0 [34] or weighted moving average filters [17] are used to get rid of noise. Gait cycles can have a different length because the user (the carrier of the smartphone) can walk at different speed levels. To ensure that every gait cycle which is extracted from the repetitive signal has the same length the feature vectors need to be normalized.

Some gait recognition approaches extract multiple outer contours of people from an image sequence and use those contours as multiple image features. In this special approach the width of the outer contour of the person is used as image feature [37]. During the gait cycle it is possible to identify specific stances that are generic, in the sense that every person transits between these successive stances. Since every person has some different poses on this stances it is possible to distinguish people based on those. In addition a 'Markovian' dependence from one stance to another can be used, which means that the current stance pose is dependent on the previous pose, since structural information alone may not lead to good recognition rate [37]. Others take those silhouette sequences and add them together to one average silhouette image [45]. The silhouettes are extracted by background subtraction to detect the object in motion in an image. In the second approach the occurrence frequency of a specific body part is extracted based on the pixel intensity at this point, therefore this average silhouette image is called gait energy image (GEI) which is used as single image feature vector for recognition. Both GEI and contour image sequence approaches are mostly used on static desktop systems with only one person at a time. Mobile gait recognition approaches however work a bit different. Due to the fact that walking is a cyclic activity the extracted features should held information according to a whole gait cycles instead of a fixed time interval [34]. A gait cycle is the time interval between two successive occurrences of the same event when walking, which is shown in figure 3.2. At the specific time the heel touches

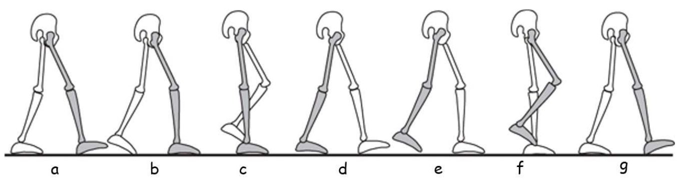


Figure 3.2: One walking cycle where the last image shows the start of a new consecutive step [35].

the ground (frame 'a' and 'g' on Figure 3.2) where the association between ground reaction force and inertial force together make a negative peak on the Z-axis (axis who is pointing to the floor) can be considered as marking point to distinguish separated gait cycles [35]. This information can be used to seperate the repetitive signal into individual gait cycles which is necessary for later on recognition.

The gait energy image approach [45] uses the obtained gait energy image as feature for recognition. By calculating the feature distance between the labeled template GEIs and the query GEI the system can obtain that the template with the smallest distance is the best match and therefore the wanted person. To archive a proper recognition the approach where HMM is used for recognition based on stance transitions a HMM for each person has to be trained with several gait cycles as features [37, 49]. The hidden Markov models are generated by using the width vector derived from the features of several gait cycles of the person and recognition is basically performed on the transitions between stances rather on the stances itself. The actual recognition is performed by evaluating the probability that a given observation sequence was generated by a HMM model from the model database. If so the user corresponding to the model from the database is the user corresponding to the given input observation sequence. Since time-information is available in the the gait signal of the mobile approach it is possible to transform the sensor signal from the frequency domain to the time domain to obtain the best features of both domains. Some features subset selection algorithms are applied to obtain the best features based on the accuracy criterion of the learning algorithm which are later on used for classification [35]. This approach uses SVM for classification using features from time and features from frequency domain. The approach of Nickel et. al. [50] which uses the authentication module represented in another paper by Witte and Nickel [76] classifies as follows. The system collects accelerometer data during the lock screen phase of the Android smartphone and save those into a database. When users want to unlock their Android smartphone the data of the last 30 seconds is used to extract gait cycles and use those to verify the user. The current extracted gait cycles will be compared with saved reference data. Reference data will be generated when a new user account gets created. If the authentication result is true the smartphone will be unlocked if the results is false the user has the chance to verify by entering the PIN or unlock pattern.

3.3 Continuous Face Authentication for Mobile and Desktop

Facial images can be captured by many different camera sensors. Some approaches work passively without forcing the user to interact with the system actively. This kind of approaches can be used e.g. for continuous authentication which is described in detail in this section. The progress in real time face detection algorithms allows face authentication systems not only on desktop PCs even low-end mobile devices with built-in cameras can use face authentication systems. Faces are usually used as physiological biometric in active entry-point verification systems [18, 66] but the face can be used as continuous authentication technique as well [55]. In continuous face authentication approaches the system captures samples continuously using a continuous real-time image stream or prerecorded image sequences rather than just a few static images like it is done in active authentication systems. Usual face authentication systems use steps like face data recording, image preprocessing, feature extraction and classification / authentication. Those steps will be discussed in detail in the following sections. In the following section classifiers like SVM, Viola and Jones, models like HMM as well as preprocessing techniques like PCA are mentioned which will be explained in detail later on in section 4.2.1, section 4.1.2, section 4.3 and section 4.4.

3.3.1 Data Acquisition

In some approaches the built-in webcam is used to acquire a video stream, the memory in the computer is used to save the images in a video frame buffer for further processing and calculation purpose [70]. Both mobile devices [1, 14, 21, 29] and desktop PCs [4, 41, 79] can use the built-in front camera or webcam to acquire 2D data. Some other approaches using external RGB-D cameras like the Microsoft Kinect sensor to acquire 3D images with depth information since 3D outperform 2D in many aspects [63], however 3D cameras are rarely shipped with off-the-shelf mobile devices. Images with 3D information are more robust against different poses, 3D images can be captured in a wide range of lighting conditions and 3D data allows a better classification between foreground and background objects. RGB-D data contains next to the RGB color values also depth information. This information can be acquired by emitting an infrared pattern which is captured by an infrared camera [32]. The major drawback of 3D face authentication is the high computational cost. Acquiring and processing 2D data however is way faster than 3D data. When we differentiate between mobile and desktop authentication one important core aspect is battery. With high computational costs the device has to calculate a lot which requires a vast amount of battery on mobile devices. This is less of an issue on desktop systems. Also observation times differentiate between desktop and mobile approaches. Whilst in desktop applications the camera sensor can nearly observe the whole time, the built in front camera of mobile devices needs some downtime as well [14]. Otherwise the battery of the smartphone would be discharged quickly and nobody would use this kind of authentication mechanism.

3.3.2 Face Detection

The face detection part of the continuous face authentication system takes an image as input and determine whether or not there are any faces in it. If so, the output of this system part is the image location and extent of each detected face. A detailed explanation on how face detection works will be given in section 4.1. Base for most face detection in real time applications is Viola and Jones's [72] ground-breaking algorithm which uses Adaboost, cascade structures and simple feature extraction. Each frame is searched for faces using Viola and Jones's algorithm. When using continuous authentication each frame or a subset frames of the video or real-time image stream is extracted. To get good results the face detection should work on faces with in-plane and out-of-plane head rotations as shown in figure 3.3. Viola and Jones's algorithm is robust and give good detection results against this rotations. To improve the face detection success rate some approaches uses

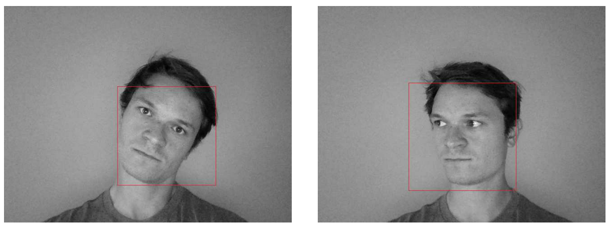

(a) (b)

Figure 3.3: (a) in-plane rotation (b) out-of-plane rotation

additional features like eye, mouth or nose detection [4, 21, 79]. If one or more of those features are found inside the face detection rectangle it is more likely to be a face. If those landmarks are close to the border or on the outside of the face bounding box, the samples will be marked as non-faces. The face detection algorithm should perform fast without long computation time, therefore errors may occur during face detection. Those regions marked as faces which are actually no faces (false positive samples) are the result of wrong face detections and the tradeoff between accuracy and computation time. In mobile applications where the background can possibly change every few frames it is more likely that the face detector finds an object somewhere in the background which will be mistakenly detected as face. In static desktop environments however the background varies not as much each session and the face detection algorithm and its parameters can be chosen based on this specific setting. Static desktop environments can be parametrized better for the specific setting which is impossible for mobile and laptop environments. To minimize the false positive detections some approaches require that the face has a minimum size [21]. When using the front camera of a smartphone the minimum height and width of the detected face has to be for example a third of the entire image to be marked as valid face. The minimum face size is depending on the smartphones camera and the detection system. This simple step can be implemented to prevent false positive matches without high computational costs.

3.3.3 Image Preprocessing

The image preprocessing step in continuous face authentication systems prepares single images or a sequence of images for later on recognition and classification. This step should reduce the computational effort and get rid of unnecessary image information. Most of the face recognition approaches work with gray scale images. This is done to reduce the information of the image and get faster calculation results from the face recognition system [38]. Indeed, color may be beneficial in some applications but the additional unnecessary information would increase the amount of training data needed to achieve good performance. Therefore all the samples get converted from RGB to gray scale. Another step to reduce unnecessary information is to scale all the faces to a fixed size. When we assume that we have an image of 64 x 64 pixel the amount of pixel features in the image is reduced to 4096. Beside feature reduction, rescaling of the face sample has the effect that all the faces have a normalized size which makes it easier for the face recognition system due to the fact that most classification models perform better on a uniform amount of features [27]. In face recognition systems the detection and recognition performance is strongly affected by a variation of illumination. Some approaches apply histogram equalization to reduce the variations of illumination (see in figure 3.4) [1, 4, 21]. Dark images will be brighten up and bright images will be dimmed. Another variation to

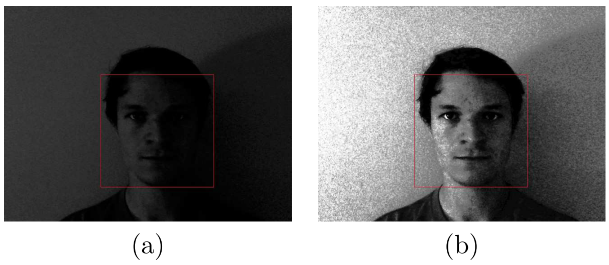


Figure 3.4: (a) Image before histogram equalization. (b) Image after histogram equalization.

encounter illumination variations is photometric normalization [79]. Homomorphic filtering is used to weaken the effects of shadows and specularity on the face by simultaneously normalize the brightness across an image and increase its contrast [16, 46]. The resulting face images are then used for feature extraction.

3.3.4 Feature Extraction and Face Recognition

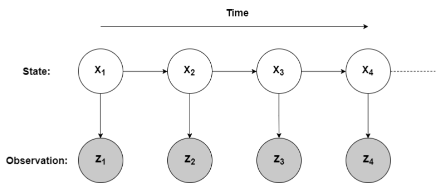
Feature extraction is animportant step where values (features) are derived from an initial set of measured data. Feature extraction will give values or vectors of values which can be later on used for recognition and classification purpose. Those features can be e.g. pixels intensity values obtained from the rescaled, gray scale image or other information extracted from those pixels. In addition to the intensity values of each gray scale pixel other more specific areas inside the face bounding box can be used. For example the intensity values within the bounding boxes of the mouth, the nose and both eyes can form a feature vector with a specific size. The mentioned bounding boxes need to be resized before hand to get normalized vectors across each face sample [21]. Some approaches divide the image into blocks, extract histograms of each block and concatenate the histogram values into a feature vector of a fixed size. For example: an image is divided into 9 blocks and the histograms have 59 bins each which leads to a vector with the size of 531 (59×9) [29]. The face identity is verified by computing the histogram intersection distance between the current face sample and a template face of the database. If the distance is below a certain threshold the face will be rejected, or reported as positive match otherwise [29]. Another way to distinguish between different faces is using measurements and distances between facial features in the human face. Those measurements can consist of the size of eyes, nose, mouth, and eyebrows and their relative positions, the distance between the ears, and the shape of the chin which form feature vectors for later recognition [4].

Some approaches use the PCA to decrease the computation time for face recognition by reducing the used features [4, 44, 60, 70]. PCA tries to find the components with the best information (eigenvectors / principal components) of a dataset [71]. It reduces the features by mapping high-dimensional data onto lower dimensions of feature spaces that contain most of the feature information [4].

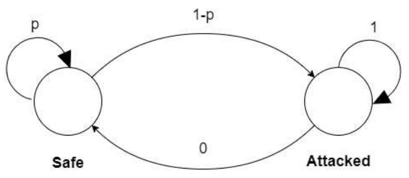
For recognition some approaches calculating the distance of extracted features to compare the input image to all images in a database [4, 70]. The input image will be recognized as the registered user with the shortest distance from the database. Other approaches using SVM to classifier and recognize input images [1]. A more detailed description about recognition and face recognition will be explained later(see section 4.2).

3.3.5 Time Function and State Storage

When using authentication systems in the continuous domain some approaches need a current state. The states could be e.g. $\{safe, attacked\}$ and depending on the input the system is in one of those states. Most continuous authentication systems use some kind of weighting over time to give the new observation sample some weight in comparison to current state [63]. The current state is dependent on all previous states and observations. Usually to realize this all past values must be stored somewhere which would be consuming an infinite amount of memory over time. To encounter this problem a Markov chain can be used [44]. A simple explanation of the model used in this approach would be that only the last state in combination of the new observation value is needed to calculate a new state. Figure 3.5 (a) shows how this Hidden Markov Model (HMM) can be visualized [65]. The state transition diagram of figure 3.5 (b) shows that once the system is in $\{attacked\}$ state, the system remains in that state and never transitions back to $\{safe\}$. A weight function is used to weight the importance of the new observation in the first place but is also used to give the current state a decay. This should prevent that the current state is valid forever when no new samples are observed. To archive those two criteria the weight function



(a)



(b)

Figure 3.5: (a) HMM with states X_t and observations Z_t over time t. (b) Shows the state transition diagram for the HMM (adapted from [65]).

$p = e^{k\Delta t}$ is defined, where Δt is the elapsed time between current time and last observation and k is a free parameter which defines the decay rate [65]. A low k for example $k = 0$ means that the user wont be attacked while a very large k value indicates that attacks are very likely.

Another approach by Crouse et al. [14] uses a simple value S_{login} which represents the confidence of the users identity. This value gets initialized to 1.0 when the user actively logs in and gets continuously updated with a given cubic transformation function f_{map} when a new face gets observed (see in equation 3.1), where the scores at individual false acceptance rates (FAR) are calculated from an independent dataset and S_{face} is the face matching score.

$$f_{map} = \begin{cases} 0.4 & FAR(S_{face}) \leq 1\% \\ f'(S_{face}) & 1\% < FAR(S_{face}) < 20\% \\ -0.4 & FAR(S_{face}) \geq 20\% \end{cases} \tag{3.1}$$

The confidence value S_{login} should not only be updated between observation times, it should be decreased in addition over time based on an integral of function f_{dec} (see in equation 3.2), where t_{logout} defines the time the device should take to log the user out if no face match data is obtained.

$$t_{switch} = 1.5 * t_{logout} - 6, \qquad f_{dec}(t) = \begin{cases} -\frac{-0.1}{t_{switch}^2} t^2 & t < t_{switch} \\ -0.1 & t \geq t_{switch} \end{cases} \tag{3.2}$$

The function f_{dec} is rather constant or can vary with time since last login. If the confidence value S_{login} ever drops below a certain threshold the user gets immediately logged out. Which leads to an equation for S_{login} (see in equation 3.3), where S_{face} is the face matching score, t_{ses} is the current time, t_{prev} is time of the previous calculation and f_{dec} and f_{map} are the functions of equation 3.1 and equation 3.2 [14].

$$S_{login}(t_{ses}) = S_{login}(t_{prev}) + \int_{t_{prev}}^{t_{ses}} f_{dec} dt + f_{map}(S_{face}) \tag{3.3}$$

The results in this approach show that in over 96% of the trails users did not get logged out during the sessions where only images of the genuine users where used. In over 89% of the trails an imposter had access for less than two minutes where the majority off all trails had less than one minute when the t_{logout} was set to 10 minutes. The evaluation used images of ten participants and captured in total 250000 images from about 3600 sessions over a period of 1-6 weeks.

Another similar approach also generates a confidence value $trust(t)$ depending on acquired data and the elapsed time between observations [9, 62]. If this confidence score drops below a certain threshold the session will be closed and the user is no longer verified. The outstanding part of this approach is that multiple biometrics can be used which will be weighted and merged before hand to generate a trust value. If at a specific time T_i at most only one biometric observation is valid (e.g. only one of multiple biometrics could be aquired) the confidence value will be calculated (see in equation 3.4), where k and s are predefined parameters to tune the decreasing function and Δt_i is the elapsed time between the last and the current observation time. This decreasing function pursues the same purpose as the decay functions from previous approaches.

$$trust(t_i) = \frac{(-arctan((\Delta t_i - s) \cdot k) + \frac{\pi}{2}) \cdot trust(t_{i-1})}{-arctan(-s \cdot k) + \frac{\pi}{2}} \tag{3.4}$$

3.4 Summary

In the above sections we have given a short overview of state of the art continuous authentication. There is a significant amount of biometrics which would fit in those sections but we only focused on the following few: keystroke/-touch pattern, gait pattern and face authentication systems which have some similarities to our approach.

In this kind of authentication systems users are not forced to use the authentication technique on purpose, users get authenticated passively in a nonintrusive way. Just by using the keyboard or touchscreen, just by walking or by looking in the front camera the approach is able to recognize, verify and authenticate or reject the claimant.

Data preprocessing is one very important part to get good results in the authentication domain. Samples which are used for recognition and authentication should be normalized to get samples of the same size. The length of the gait signal or the height of the gait energy image or face image should be the same among the captured samples for a better classification. Another part of data preprocessing can be noise reduction. This could be done using filter mechanisms, e.g. filters on gait signals to reduce noise due to the fact that accelerometer sensors are prone against interferences. Also steps to emphasize the actual payload can be applied, e.g. like histogram equalization on images in order to increase the contrast in images with poor lighting conditions. Some other approaches are very vulnerable against outliers, e.g. like keystroke pattern, where long pauses between typed keys may harm the robustness of the model. In this kind of approaches these outliers need to get removed before the actual recognition and authentication. All in all the preprocessing of data is very important to get good recognition results in the end.

In terms of feature derivation PCA is a good tool to reduce the amount of features which will be later on used for classification. Reducing the features will lead to lower computation times and lower storage in the memory with nearly as good recognition results as using all original features. Our approach however do not use principal component analysis it only reduces the features by rescaling the image to a smaller size.

Those kind of authentication features need somehow consider time as constant. Where some approaches uses fixed or varying time windows to check if the current user is authorized to enter the system, other approaches check from time to time and use this observation data to increase or decrease a confidence value. If this value drops below a certain threshold the system transits into a specific state where users are limited in their possibilities or even getting locked out of the system. If there is such a state which declares the users identity (owner or attacker) the state needs to change over time as well when no new observations come into the system. This needs to be done to prevent attacks where the imposter simply interrupt the input (e.g. hiding the camera in a face authentication system). Some systems have a predefined threshold to define a minimum required input to prevent the mentioned problem (e.g. define a minimum amount of typed keys in keystroke pattern authentication). Other approaches use some kind of a decay function, where the confidence value drops over time when no new examples get acquired. Both approaches will lead the system to transit in attack-state when no new samples get observed.

The mentioned biometric continuous authentication systems uses different approaches to acquire one common goal: creating a system which is safe and hard to attack in one hand but nonintrusive and continuous on the other.

Chapter 4

Building Blocks

This chapter will give some general information about face detection. Classifiers like Support Vector Machines (SVM) and K-nearest-neighbors (kNN) as well as tools like hidden Markov models (HMM) and principal component analysis (PCA) are described. These parts are necessary in order to understand some techniques which are mentioned in the chapter 3 related work and chapter 5 about our approach.

4.1 Face Detection

Face detection is one of the most studied and researched computer vision topics [82]. It is one of the fundamental techniques for human-computer interaction (HCI) and the step stone to all facial analysis algorithms like, face alignment, face modeling, face relighting, face recognition, face verification, face authentication, head pose tracking, facial expression tracking, gender recognition, and many more. The basic task is simple: determine whether or not there are any faces in an arbitrary image. If so, return the image location and extent of each detected face. This might be easy for humans but can be very challenging for machines due to the following factors: pose, presence or absence of structural components, different facial expression, occlusion, image orientation and imaging conditions [80]. As we can see there is a variety of factors which can affect the face detection task and make this a hard achievement in some applications. But the increasing computational power and the non stopping evolution of algorithms makes face detection more feasible in the real-world and adapt it to more applications. Applications like Facebook uses face detection mechanisms for automated person tagging [81]. Furthermore, the majority of commercial digital cameras uses embedded face detectors to help auto-focusing. This distribution of the face detection usage was especially archived by Viola and Jones (explained in more detail in section 4.1.2), who made face detection practical in nowadays real-world use-cases.

4.1.1 HAAR Cascade Classifier

The base for HAAR cascade classifiers are the HAAR-like features [75]. HAAR features do not use intensity in pixels (0 - 255 in gray scale images). They rather use the change in contrast between adjacent rectangular groups of pixels, to determine relative dark and light areas. HAAR features are groups of two, three or even four rectangles which can look like the examples in figure 4.1 [73]. These HAAR features can be easily scaled by adjusting

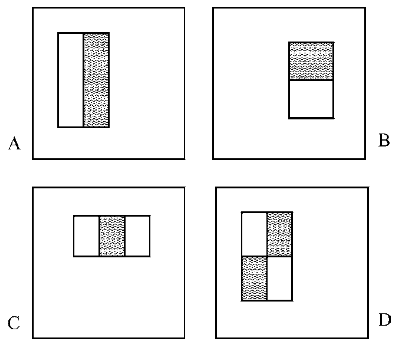


Figure 4.1: A and B shows two-rectangle feature, C shows a three-rectangle feature and D a four-rectangle feature [73].

the size of the rectangles. This allows object detection with scale invariance.

Integral Images

The rectangular features of an image can be calculated by so called integral images which are the intermediate representation of the image [73]. In other words, integral image at location x, y contains the sum of pixels' intensity values (input image) to the left and above x, y including x, y. Let $I\left[x, y\right]$ be the original image and $II\left[x, y\right]$ be the integral image. The integral image gets calculated like in equation 4.1.

$$II(x,y) = \sum_{x' \leq x, y' \leq y} I(x', y') \tag{4.1}$$

Each rectangle of the HAAR feature is calculated by those integral images. The HAAR feature itself gets calculated by the sum of the pixels which lie within the white rectangles subtracted from the sum of pixels within the gray rectangles. The integral image allows a rapid calculation of any rectangular sum because only four array references are needed seen in figure 4.2 of [64] and corresponding equation for calculating input image pixel with equation 4.2 and calculating integral image value with equation 4.3 obtained from [64]. This is the base of the fast performance of HAAR-like features because the integral image can be computed in one pass through

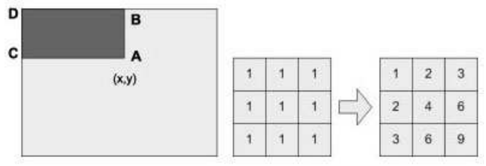


Figure 4.2: Calculation of locations in the integral image. The mid image is the original image $I(x, y)$, the most right image is the integral image $II(x, y)$ [64].

the image.

$$I(x', y') = II(x, y) + II(x - 1, y - 1) - II(x - 1, y) - II(x, y - 1) \quad (4.2)$$

$$II(x, y) = I(x', y') - II(x - 1, y - 1) + II(x - 1, y) + II(x, y - 1) \quad (4.3)$$

4.1.2 Viola Jones Face Detection Algorithm

As already mentioned above Viola and Jones created one of the most used and well known face detection systems [72]. This approach classifies objects based on simple features instead on pixel basis. The simple reason therefor is that classification on simple features is much faster than classification on pixel base. The features which are used are based on HAAR-like features. To be more specific three different kinds of HAAR features are used, two-rectangle feature, three-rectangle feature, four-rectangle feature which are already described in Section 4.1.1. After the calculation of the integral images of each rectangle feature an algorithm based on AdaBoost is used. This will select a small number of critical visual features from a larger set. AdaBoost is a machine learning algorithm which is implemented to only find the best features. AdaBoost will beside the feature reduction purpose also boost the classification performance of a simple (weak) learning algorithm. Some simple classifiers are used to reject the majority of sub-windows before more complex classifiers are called to find the object of interest. In its original form AdaBoost uses multiple weak classifiers and combine those to one strong classifier which allows background regions of the image to be quickly discarded while spending more computation on promising face-like regions.

4.2 Classification and Face Recognition

4.2.1 Support Vector Machines

Support vector machines (SVM) are large margin classifiers. The goal for SVMs is to seperate the data samples with a hyperplane where the margin between the hyperplane and the samples is as large as possible on all sides [12]. SVM try to find a hyperplane that is consistent with the data while committing least to it. Figure 4.3 shows a linear classification example of two classes (white and black dots) [23]. SVM using a hyperplane (the solid

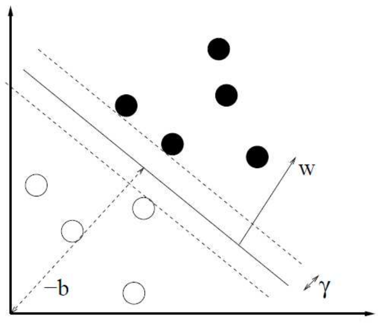


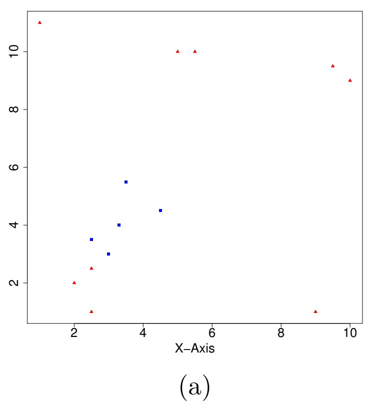
Figure 4.3: Classification of two classes (white and black dots) [23].

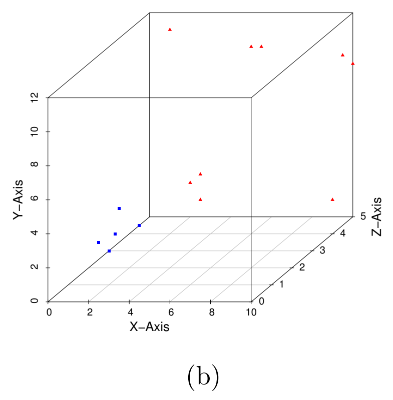
line in Figure 4.3) to seperate the samples where the margin is as large as possible for the samples on all sides. There are two additional hyperplanes in figure 4.3 which define the margin between the classes of the sample data (dashed lines). Those margin hyperplanes are directly located on some of the samples which are nearest to the classifier hyperplane (the most outer samples). The samples on the margin-hyperplanes are called support vectors and those are necessary to create the classification hyperplane. So basically SVM is an algorithm that finds those specific samples (support vectors) to create the best separating hyperplane with the highest margin between the classes. The optimization problem of finding this maximum margins is a quadratic programming problem [12].

Support vector machines solve the separation of the classes with a linear function. If the classification is not able to be done via a linear function the kernel trick comes into play. With the kernel trick the system projects the data into a higher dimensional feature space to make it linear separateable. The kernel itself is the function that projects the data into a higher dimension. The kernel knows how the points gets transformed into the higher dimension. The system uses those kernel-function to operate in those higher dimensions. Therefor it is not necessary to do the computation that actually transform the points into that higher dimensional space. The fact that the actual complex computation has not to be done makes SVM so fast [23] [12]. Figure 4.4 will visualize an example for applying the kernel-function.

4.2.2 k-Nearest-Neighbor(KNN)

K-nearest-neighbor (KNN) is a fundamental and simple classification method. Basic idea for the classification is to select the class of an unknown sample based on the the k-nearest-neighbors among a dataset of already classified





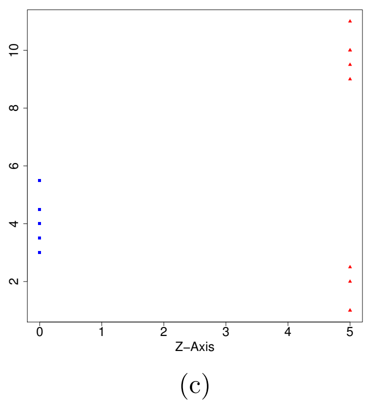


Figure 4.4: (a) The samples in 2D space are not linear separate-able. (b) The samples with additional dimension. (c) The samples seen from different view are linear seperate-able.

training samples [57]. The distance between the unknown sample and its neighbors is commonly calculated by the Euclidean distance. Let x_i be an unknown input sample with a total number of p features $(x_{i1}, x_{i2}, ..., x_{ip})$. The Euclidean distance between sample x_i and sample x_1 is defined as seen in equation 4.4.

$$d(x_i, x_1) = \sqrt{(x_{i1} - x_{11})^2 + (x_{i2} - x_{12})^2 + ... + (x_{ip} - x_{1p})^2} \tag{4.4}$$

A majority vote among the k samples with the lowest distance to the unknown sample will decide the class of the sample. Figure 4.5 shows an example where the parameter k is defined as 4 and the samples have a total number of $p = 2$ features. The unknown green sample will be assigned to class ‘A‘ because among the 4 nearest neighbors of the test samples, the most frequent class is ‘A‘.

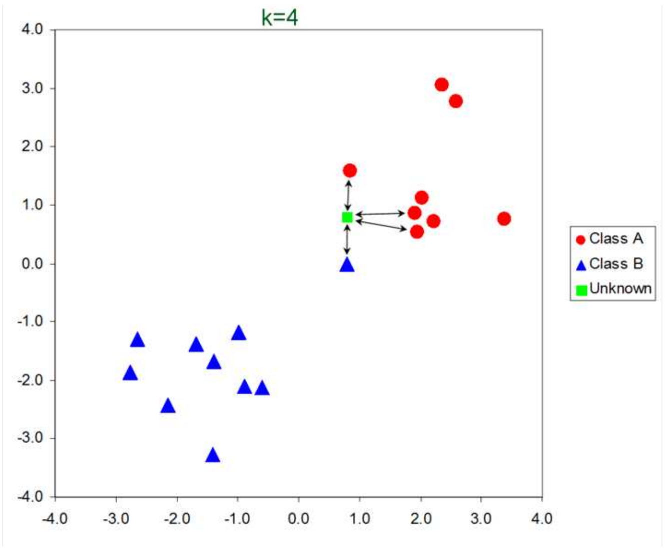


Figure 4.5: Among the 4 nearest neighbors of the test sample, the most frequent class label color is red [57].

4.3 Hidden Markov Model

Hidden Markov models (HMM) are a tool for representing probability distributions over sequences of observations or for modeling time series data. HMM assume that an observation at a specific time t was generated by some stochastic process whose state S_t is hidden from the observer [26, 58]. A stochastic process is defined as a collection of random variables associated with or indexed by a set of numbers usually viewed as points in time [39]. The hidden state also has to satisfies the Markov property which implies that the current state S_t is independent of all previous states. Which means that the state at some point in time encapsulates everything the system needs to know about the history of the process in order to predict the future of the process [26, 58]. These two properties are therefor the reason for the naming of the model. Further details and examples about HMM are explained in the publication of Rabiner et. al. [58] and the publication of Ghahramani et. al. [26]

4.4 Principal Component Analysis

The principal component analysis (PCA) tries to find the components with the best information (eigenvectors / principal components) of a dataset [71]. It reduces the features by mapping high-dimensional data onto lower dimensions of feature spaces that contain most of the feature information of a dataset [4]. PCA works on all kind of data and is used in different approaches as seen in the related work chapter 3. This section especially points out the usage of PCA on face data. The PCA takes sample images with the same size e.g. k x k-sized face images where k is the amount of pixels (PCA works on rectangular images as well). Each face sample will be represented as an unfold 2D vector with k^2 values. Equivalent a face is one point in a k^2-dimensional space. PCA creates a matrix of out of those vectors which is $k^2 \times N$ where N is the number of face samples used for the PCA. Afterwards the mean-vector (average face vector) will be subtracted of each face-vector and the covariance matrix will be calculated. This has to be done to get eigenvectors (principal components) which are in the so called principal component (PC) space. The covariance matrix is a matrix with eigenvalues as diagonal values. Eigenvalues also gets calculated by the PCA. Eigenvalues indicate how much the eigenvector varies from the mean-vector (average face). The first eigenvector (with the highest eigenvalue) shows the most prominent deviation from the mean face. This is one dimension in the k^2-dimensional space. On this dimension (eigenvector) all sample faces vary the most from the mean. Eigenvectors can be graphically represented as so called eigenfaces by transforming the eigenvector back to original space. Any face used in the PCA can be represented by a linear combination of eigenvectors (fold and transformed back to original space) and the corresponding weights. The weight indicates how much influence an eigenvector has on the representation of a specific face. All face samples are using the same principal components / eigenvectors. The only variations among the faces are the corresponding weights (projection coordinates of the PC-space). This means if 40 eigenvectors are needed to represent the faces properly only a vector with 40 values (40 weight values) is needed for the face classification model. As a result of this each face sample can be represented by its best individual PCs depending on the intended variance in original space. Additional information about PCA are explained in more detail in the book of Dunteman [19] and a publication of Turk and Pentland [71].

Chapter 5

Our Approach: Continuous Mobile Face Authentication

In this chapter we present a continuous mobile face authentication approach prototype which distinguishes between the owner of a smartphone and other users (possible attackers) which might want harm the owner. There are plenty of other face authentication approaches on the market or in research but most of them are explicit, some exceptions are stated in the previous related work section 3. Explicit means that users have to explicitly look into the camera to be authenticated. This demands user attention and time, so the user experience is negatively affected by those points. The main motivation for this face authentication approach is to create an unobtrusive, continuous and mobile face authentication prototype. Users need not actively authenticate on the system, they work on the device, and automatically remain authenticated when they have the permission to. We summarize the design goals for such a system as the following:

- Continuous: the current user of the smartphone will be validated all the time instead of only at the entry-point.
- Mobile: the use cases for the system are designed for smartphones and the smartphones front facing camera.
- Unobtrusive: the authentication is transparent for users and therefore no active interaction with the authentication system is necessary.

In our approach we only consider the authentication of attackers against one smartphone owner, because a smartphone is usually privately owned and not shared by multiple users. The continuous mobile face authentication system requires video-sequences or camera streams captured by the front facing camera and use them for authentication.

5.1 Continuous Mobile Face Authentication Toolchain

The following section will give an overview of the used modules in the system. First part of our proposed continuous mobile face authentication toolchain (see figure 5.1) is the acquisition of data by using the front facing camera of a smartphone to acquire video sequences. In the next step single frames get extracted and forwarded to the face detection module. For face detection, we use OpenCV's[1] HAAR feature-based cascade classifier based on Viola and Jones' ground breaking algorithm [52] to detect faces in the extracted images (see section 4.1.2). After the detection of the face region the image gets preprocessed by segmenting the face region, applying gray scale filters to get rid of color information and histogram filters to increase the contrast in the image. Another part of the preprocessing module is the rescaling process of the image to a fixed size of 50 x 50 pixel. This is important to reduce the features for decreasing calculation time for recognition and to get a normalized image size among all upcoming samples. The next important module creates difference images out of the segmented and preprocessed face images. This is important for the recognition module being able to handle face images which are not in the training dataset. The difference images can be either positive when both input images are of the smartphone owner, or negative otherwise. Based on difference images we apply face recognition using the LibSVM[2] library [10]. We first train a face classification model with the preprocessed and labeled difference images then perform face recognition to get a classification result. The system has two possible states which indicates if the system is currently recognizing the owner of the smartphone or an allegedly attacker (see transition diagram in related work section 3.5). Those states can change over time and are depending on the current confidence score. In order to get a confidence score a confidence value will be calculated as a combination of the recognition probability, the old confidence value, a weight function and a time decay function. The weight function is needed to give the observation samples a weighting depending on the elapsed time between two consecutive observations. The time decay function decreases the confidence score over time to avoid attacks. For the owner enrollment phase some reference images are needed to define who is the owner of the smartphone. After the classifier model creation video-sequences or live video-streams can be used to predict on.

[1]OpenCV is an open source computer vision library. goto OpenCV homepage.

[2]LibSVM is an integrated software for support vector classification, regression and distribution estimation. goto LibSVM homepage.

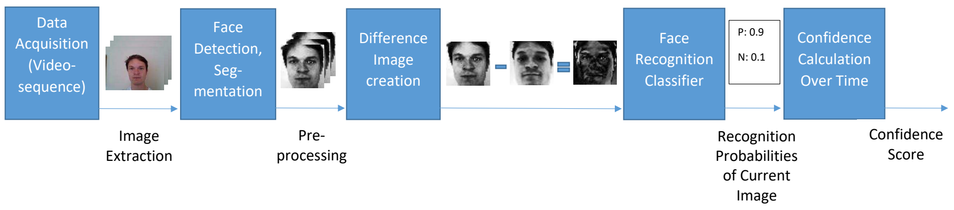

Data Acquisition (Video-sequence)
Image Extraction
Face Detection, Seg-mentation
Pre-processing
Difference Image creation
Face Recognition Classifier
P: 0.9
N: 0.1
Recognition Probabilities of Current Image
Confidence Calculation Over Time
Confidence Score

Figure 5.1: Overview of modules used in the continuous mobile face authentication system.

5.2 Data Acquisition and Image Preprocessing

In our approach the data for face classification model training and recognition are collected via the front facing camera of a smartphone. The collected data samples are mp4-video sequences with a bit depth of 24 bit, a resolution of 640 x 480 pixel, a frame rate of 30 fps and a length of 60 seconds. The first step in the face detection module is to extract single frames from the video sequence. In our approach every frame of the video sequence is used for recognition purpose. After one frame is extracted the face will be detected by the HAAR feature-based cascade classifier of the OpenCV library. To search for the face in the whole image one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily resized in order to be able to find the objects of interest at different sizes. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales [52]. After a face is detected the location and the dimension of each face will be returned. This will lead to a list of possible faces because the classifier is able to detect more faces in one image. Due to the fact that smartphones are on different locations with different lighting conditions it is possible that the classifier predicts non-face objects in the image as face. Therefor only the biggest face region will count as face when the size of the region is at least 25% of the whole image. This will prevent that people in the background (shoulder surfing, see figure 5.2) or other small objects in the image falsely count as faces.

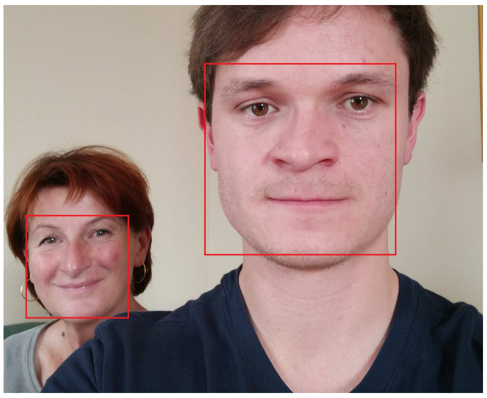


Figure 5.2: Owner of the smartphone with a shoulder surfing person in the back.

After the detection of the face region of interest (at least 25%) the region gets segmented and used for further preprocessing. The next step is to apply a gray scale filter to the color image. This will reduce the file size (approximately from raw: 360KB to gray 208 KB) as well as the information per pixel (from raw: 3 channel to gray with 1 channel). Color may be beneficial in some applications but the additional unnecessary information would increase the amount of training data needed to achieve good performance [38]. The next step is to apply a histogram filter to increase the contrast in gray scale images with bad lighting conditions. Figure 5.3 shows the difference between an image with bad lighting and the same image with applied histogram filter. The last step in the preprocessing module is image

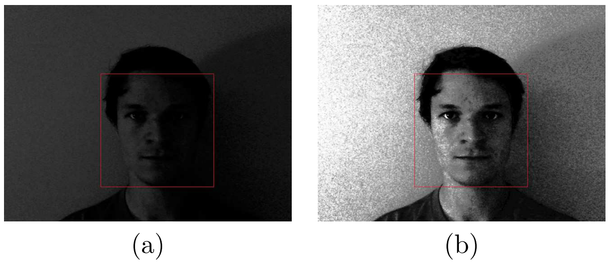
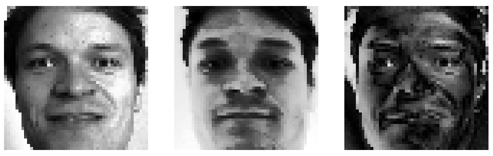


Figure 5.3: (a) Image before histogram equalization. (b) Image after histogram equalization.

normalization. Face regions in the image can range from minimum 25% (of 640 x 480 pixel) to a maximum of 640 x 480 pixel (input image size). To get a normalized image size among all image samples for training as well as those image samples for later on recognition, the face region will be resized to 50 x 50 pixels. This makes it easier for the face recognition system due to the fact that most classification models perform better on a uniform amount of features (pixels). The computation time needed for classification also will benefit of the feature (pixel) reduction. Another type of feature reduction would be PCA which is done in some approaches of the related work in section 3.2 and section 3.3 but omitted in our approach.

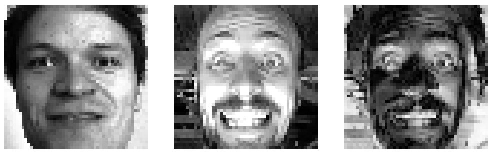
5.3 Difference Image Creation

A very important part of the toolchain is the creation of difference images. This will allow the system to work on samples of participants which are beyond the database. If only samples within the database can be recognized correctly, the authentication system is called gallery dependent which is a big problem in real world applications. The system could not handle samples of people outside of the database. For example: if users are not in the database they will be matched as one person which is in the database. Users will be matched as the database sample with the highest equality. If this match would be the owner by coincidence, the attacker would get access to the system and to private data or even passwords. Difference image creation can prevent this and make the system gallery independent. To create such difference images two preprocessed images will be used. The images are represented by 2D matrices where every matrix entry is the density value of the pixel at this point. To calculate the difference image (matrix) a simple absolute difference calculation (absolute distance per feature) between both matrices will be executed. This calculation generates samples as in figure 5.4 (c) and figure 5.5 (c). In order to get a gallery independent system we want to create a two-class problem (owner and not-owner) out of the N-class problem(where N is the number of different people in the training database). Therefor before the training of the classification model the difference images need to get labeled with N for not-owner samples (negative) and P for owner samples (positive). In training those labeled samples are used to train the classifier, in the application scenario the trained classifier will predict on unknown difference images. As we already mentioned our approach only considers the authentication of people against one smartphone owner therefor we need to define which label is corresponding to the owner first. If now a difference image is calculated with both images of the owner the sample gets labeled as positive sample (like in figure 5.4) or labeled negative (like in figure 5.5) otherwise. After the creation of a good amount of difference images the face classification model can be trained.



(a) (b) (c)

Figure 5.4: (a) Owner reference sample of the database (b) Owner observation sample (c) Difference image, labeled positive.



(a) (b) (c)

Figure 5.5: (a) Owner reference sample of the database (b) Not-owner observation sample (c) Difference image, labeled negative.

5.4 Recognition

Before the actual recognition part where new, unlabeled, preprocessed, difference images are going to be classified, the SVM face recognition model need to be trained. Therefor the LibSVM model training methods are used with a set of parameters [10]. Parameters which need to be considered are for linear classification SVMs the C and the γ-parameter, where C defines the smoothness of the margins and γ defines the influence of a single training sample. If the γ-value is low samples which are far away from the decision boundary (hyperplane) also take into consideration. If γ is high, only samples close to the decision boundary will be considered. A small C allows some samples between margins, a high C-parameter allows no samples between margins which can lead to overfitting.

After the successful training of the classification model with the labeled difference images we now perform face recognition. When a new, unknown and unlabeled face sample comes into the recognition module (either from a video-sequence or a camera stream) new difference samples getting generated. The difference samples will be calculated with the current observation and pre-saved reference owner face samples. This will lead to a list of N difference images, where N is the amount of pre-saved reference owner face samples. Depending on calculation speed and prediction accuracy a good amount of owner samples need to be chosen here. Recognition will be performed on that generated list of difference images to calculate probabilities related to the two classes $\{owner, not - owner\}$.

5.5 Weight Function

This module takes the probabilities of the recognition module and multiplies them with the output value of a weighting function. Depending on the time between two consecutive images out of the video sequence, the new face sample need to be weighted. This is necessary when a new face sample comes into the authentication system after a long period of no face detections. The new sample should have a big impact on the confidence score. The calculation of the new confidence score is stated in equation 5.3 in the confidence score calculation section 5.7. If the new face would not have a strong impact, the face of an allegedly attacker would not influence the confidence score that much and the attacker would have access to the smartphone for a long time (see section 6.1 scenario 3). In another case if the user wants to use the smartphone after a period where no faces got detected, he has to wait for a long time to increase the confidence score for accessing the smartphone and using the wanted features.

So basically the weight function f_{wei} is a tool to weight how much a new face sample should influence the current confidence. The function f_{wei} is an exponential function (see equation 5.1) with two parameters Δt and k, where Δt is the elapsed time between two consecutive face samples and k is a constant which is responsible for the strictness (slope) of the function. With a high k, the old confidence value has more impact than the new observation value when calculating the new confidence score. With a low k, the old confidence value has less impact than the new observation value when calculating the new confidence score (see figure 5.6).

$$f_{wei}(\Delta t) = e^{-\frac{\Delta t}{k}} \tag{5.1}$$

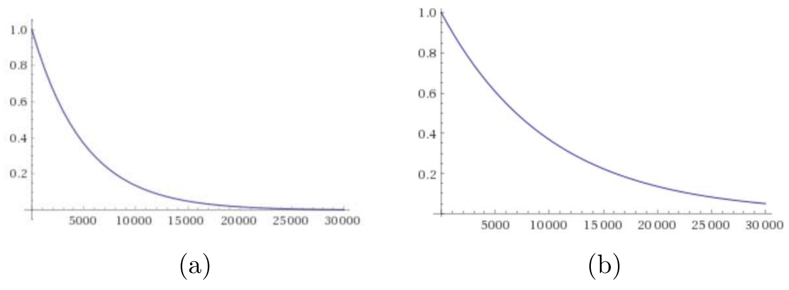


Figure 5.6: Plot of the weight function, where on the y-axis is the function value f_{wei} and on the x-axis is Δt ranging between 0 and 30000. (a) weight function with a k of 5000, (b) with a k of 10000.

5.6 Time Decay Function

One of the most important parts of our approach is the time decay function f_{dec} along side with the the weighting function of section 5.5. In a continuous system such as in our approach one important factor is time. As time goes by the current state of the system $\{safe, attack\}$ should change when the confidence score drops below a certain level. As mentioned in section 6.1 scenario 2 and scenario 4, the system should transit into *attack*-state when no new face samples are acquired for a specific period. This should prevent a simple attack approach where the allegedly attacker gets access to the smartphone just by hiding the front facing camera. Therefor, every observation loop where no new face gets detected a function gets called which calculates a time decay value which gets multiplied with the current confidence value. The function is the same as the weight function in equation 5.2 but with a different constant k depending on the desired confidence decay-duration.

$$f_{dec}(\Delta t) = e^{-\frac{\Delta t}{k}} \tag{5.2}$$

Similar to our weight and decay functions are the functions in Crouse et. al. [14] and Sim et. al. [65] continuous authentication approaches. Both as well as our approach uses functions to weight the importance of incoming samples. The confidance value is depending on the probability calculated by the classification model in combination with the weight function. Sim et. al. exponential weight function looks similar to ours, where Crouse et. al. follows a different approach and uses a cubic transformation (see equation 3.1). Sim et. al. uses a single function in each calculation iteration for two reasons: a) weighting the importance of a new observation; and b) for time decay calculation. However in our approach only one function is used each iteration. The weight function is used when a face is detected and the decay function when no face is detected. Crouse et. al. also uses two independent functions for weighting samples and for time decay calculation but both at each iteration.

5.7 Confidence Score Calculation

The confidence score which is responsible for the systems state transitions $\{safe, attack\}$ get calculated / updated in two different ways as seen in figure 5.7. The different ways are depending on the fact if a face can get detected in the extracted face image or not. If a face is detected the confidence score gets multiplied with the weight function f_{wei} (see section 5.5). If there is no face detected the confidence score should decrease over time. This is achieved by multiplying the current confidence value with the result of the time decay function f_{dec} (see section 5.6).

A face is detected. The first step for calculation the confidence score when a face gets detected is the generation of the difference images. Depending on the amount of pre-saved user samples the same amount of difference samples are going to be generated (see section 5.4). Lets say there are 20 pre-saved, preprocessed owner face samples, this will generate 20 difference face samples (20 owner samples and one observation sample). The calculation of the confidence score X_t is the average value Z_t of the 20 recognition values multiplied with the opposite weight function value $(1 - f_{wei})$ added the old confidence score X_{t-1} multiplied the weight function f_{wei}, as seen in equation 5.3.

$$X_t = (X_{t-1} \cdot f_{wei}) + (Z_t \cdot (1 - f_{wei})) \tag{5.3}$$

No face is detected. When no face can be detected in the detection module the current confidence score X_t will be decreased by multiplying the value of the time decay function f_{dec} which is seen in equation 5.4.

$$X_t = X_{t-1} \cdot f_{dec} \tag{5.4}$$

The confidence score is saved in a simplified hidden Markov model. Only the previous confidence score and the current observation are needed to calculate a new confidence score. When the confidence score drops below a certain threshold the system should transit into *attack*-state and make the system safe against an attacker. Sim et. al. used a similar approach also with a simplified HMM [65]. The main difference between their approach and ours is that once the system transits into *attack*-state it can not transit back into $safe$-state. Our approach however transits back to $safe$-state once the threshold is reached, which is necessary to achieve the unobtrusive design goal. Figure 5.7 shows the procedure of our approach.

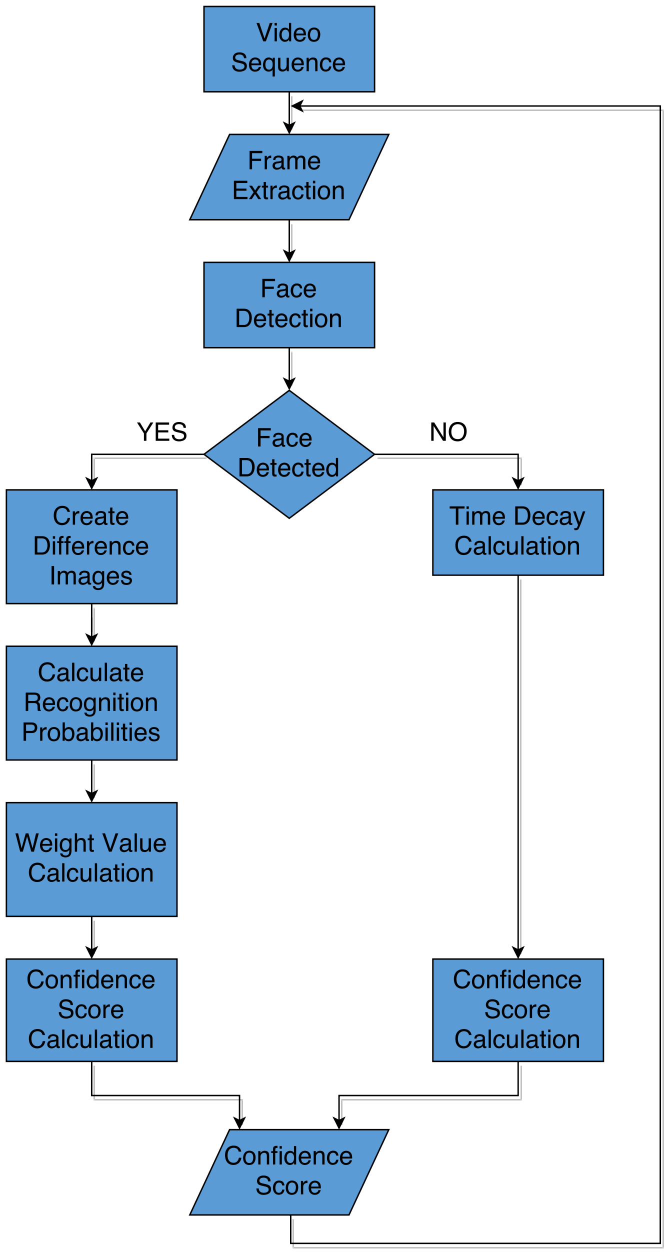


Figure 5.7: Flow chart representation of the continuous mobile face authentication system.

Chapter 6

Evaluation Data

To test and reproducibly evaluate our approach we created a video face-database. We created this face-database under different conditions in-the-field with a smartphone recording application. We decided to do this for several reasons: a) to generate videos which are authentic to general smartphone usage, in terms of mimic and gestures. The participants should act normal and pretend to use the smartphone like they usually do. No specific gestures nor mimic was predefined. To help the participants to act normal the recording application shows random articles during the recording session to distract them from the actual recording task; b) to get different ambient and lighting conditions which occur usually when interacting with the smartphone. Three main lighting conditions are defined for each participant. **Outdoor daytime** where it can happen that the sun light shines into the camera. This can create shades and darkens parts of the participants faces. **Indoor with poor lighting** conditions to get videos with darker surrounding where the face appears darker than usual and **indoor with good lighting** conditions where artificial light sources give good lighting conditions. Also the backgrounds are random and not chosen on purpose to achieve the mobile design goal; and c) for generating videos for testing the application with different scenarios. In the testing scenarios either one or two participants are involved and simulate the smartphone usage with and without an allegedly attacker. We basically came up with four scenarios to describe the interactions where the smartphone owner and an allegedly attacker are involved.

6.1 Scenarios

In this section we show some of the application scenarios to depict the importance of the weight function module and decay function module. We basically came up with four scenarios to describe the interactions where the smartphone owner and an allegedly attacker are involved.

Scenario 1: Owner or attacker only. In the first scenario there is only one participant involved: either the owner of the smartphone or an allegedly attacker. The system only obtains face samples either of the owner where it should be in $safe$-state, or of the attacker where it should be in *attack*-state, along the whole observation period.

Scenario 1: Owner or attacker only. In the first scenario there is only one participant involved: either the owner of the smartphone or an allegedly attacker. The system only obtains face samples either of the owner where it should be in $safe$-state, or of the attacker where it should be in *attack*-state, along the whole observation period.

Scenario 2: Owner or attacker with a pause. In the second scenario again only one participant is involved, but this time the observation period contains a pause where no faces are detected. In this pause the system should not be in $safe$-state. This should prevent that an attacker is able to access the smartphone just by interrupting the obtainment of new face samples by hiding the front camera. After the pause where no faces were obtained the system should react fast to new samples either of the owner or the attacker.

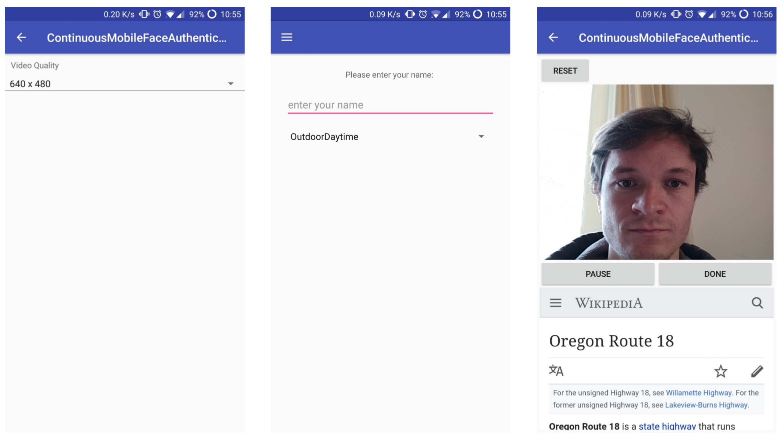
Scenario 3: Owner and attacker. In this scenario both participants, smartphone owner and allegedly attacker are involved. Samples of the owner are immediately followed by samples of the attacker. The system should be able to adapt fast and transit into *attack*-state when non-owner samples are obtained or transit into $safe$-state the other way around (attacker samples followed by owner samples).

Scenario 4: Owner and attacker with a pause. The last basic scenario is like scenario 3 but instead of the prompt alternation between the two participants a small pause is in between. During the pause the system should be in *attack*-state to prevent the threat like described in scenario 2. The system also should be able to adapt fast to the participant alternation described in scenario 3.

Those videos which are not used for testing the scenarios are basically recordings of the participants faces using the front facing camera where they interact with the smartphone. This videos have a length of 60 seconds and a resolution of either 640 x 480 pixel or 320 x 240 pixel depending on the setting used in the recording application (640 x 480 is default setting). The videos are colored RGB videos with 30 frames per second in mp4 format. The database was created in multiple sessions on random locations to achieve the mobile design goal.

6.2 Recording Application

To record the evaluation data for the authentication approach an Android recording application was developed. The application uses the front facing camera of a smartphone to record videos of the participants. The application basically has three views (see figure 6.1). In the setting screen the video resolution can be chosen. The choice is between 640 x 480 pixel or 320 x 240 pixel, where 640 x 480 pixel is the default setting. In the initial screen the participants have to enter the name and the current lighting condition. They can chose between outdoor daytime, indoor good light and indoor poor light. Depending on the chosen name an ascending index is going to be generated which is necessary to distinguish between owner and non-owner video sequences during the model training. In the recording screen a random Wikipedia article is displayed to distract the participants from the actual video recording. This should help the participants to look natural and not having a fake mimic or gesture.



Video Quality
640 x 480

Please enter your name:
enter your name
OutdoorDaytime

Oregon Route 18

For the unsigned Highway 18, see Willamette Highway. For the former unsigned Highway 18, see Lakeview-Burns Highway.

Oregon Route 18 is a state highway that runs

Figure 6.1: Application screenshots. (a) Setting screen to set the video resolution, (b) the initial screen to set the name and lighting condition, (c) the recording screen.

6.3 Participants

For the database creation 23 different participants took part of the video recording tasks. The participants are different in age and gender. Also some of the participants were wearing glasses or hats during the recording sessions. Figure 6.2 will show the distribution of the different factors like age, gender and accessories worn during recording session. Unfortunately the participants are from the same ethnic origin which is unfavorable for a proper data

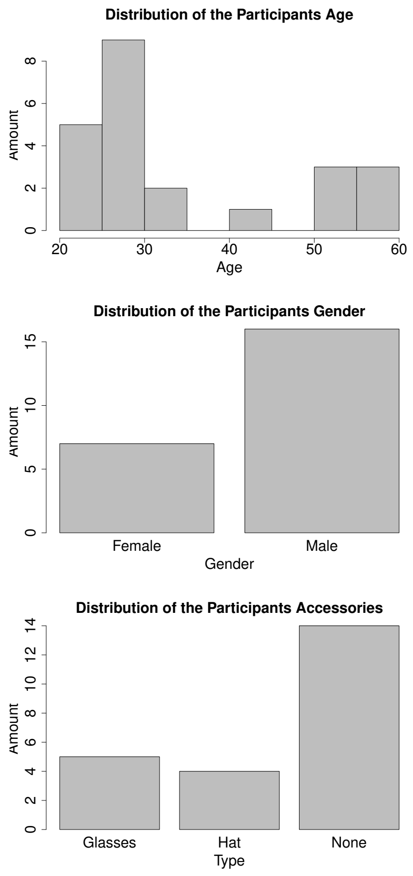


Figure 6.2: Diversity among the participants in different factors like age, gender and accessories worn during recording session.

diversity. To depict the different lighting conditions figure 6.3 will show four different images with different conditions. Image (c) and (d) of figure 6.3 are both recordings from outdoor, however in image (d) the light angle was poor so the face of the participant occurs darker than in image (c). Those are scenarios which the application must handle and therefor random locations and different lighting conditions were recorded. Figure 6.4 show some examples of images extracted from the video sequences on different locations and the corresponding preprocessed gray scale image which will be used for model training.

6.4 Test Data for Model Training

For face authentication with video-sequences or video streams first of all a classification model needs to be trained. Therefor images need to be ex-

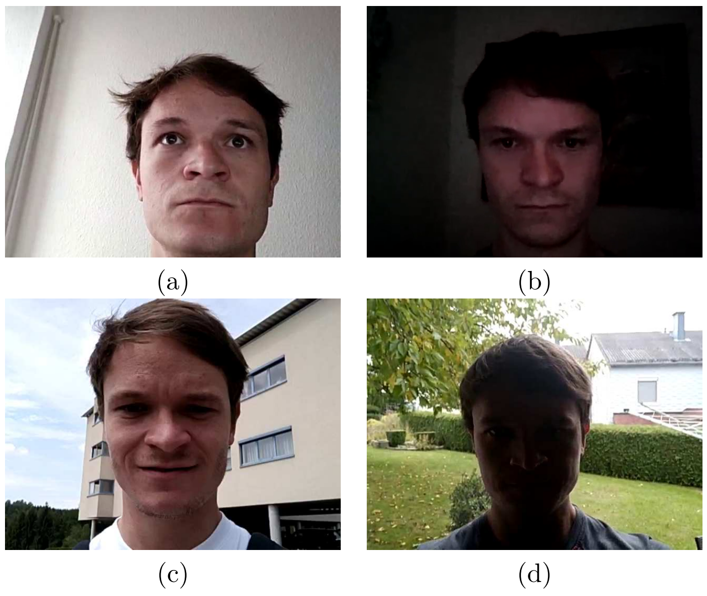


Figure 6.3: (a) indoor with good lighting conditions. (b) indoor with poor lighting conditions (c) outdoor with good lighting conditions (d) outdoor when face is full of shade.

tracted from the video-sequences. But not every frame is used of the recorded 60 second video sequences, in order to save storage every 20th frame is used when a face is detected. If no face is detected the next frame is used. When ten times in a row no face can get detected, 30 frames (a whole second of the video) are skipped and the next frame is used instead. This should avoid long computation time when multiple frames in a row do not contain a single face. This leads to an amount of approximately 50 to 80 images per video depending on the lighting and the corresponding face detections. For the creation of the face-database three different videos of 10 different persons out of 23 participants were chosen (approximately 50% of video-sequences used for training and approximately 50% of video-sequences used for evaluating the model). Unfortunately there was one video were we could not extract face images because the encoding is broken, therefor only 29 videos can be used. To get a good ratio between owner and non-owner difference samples, the predefined owner has more than just three different videos and not every frame of the non-owner videos is considered for the different image creation. A owner to non-owner ratio of approximately 1:2 was intended which results to a total amount of 12543 difference images and a good ratio between owner-class and non-owner-class.

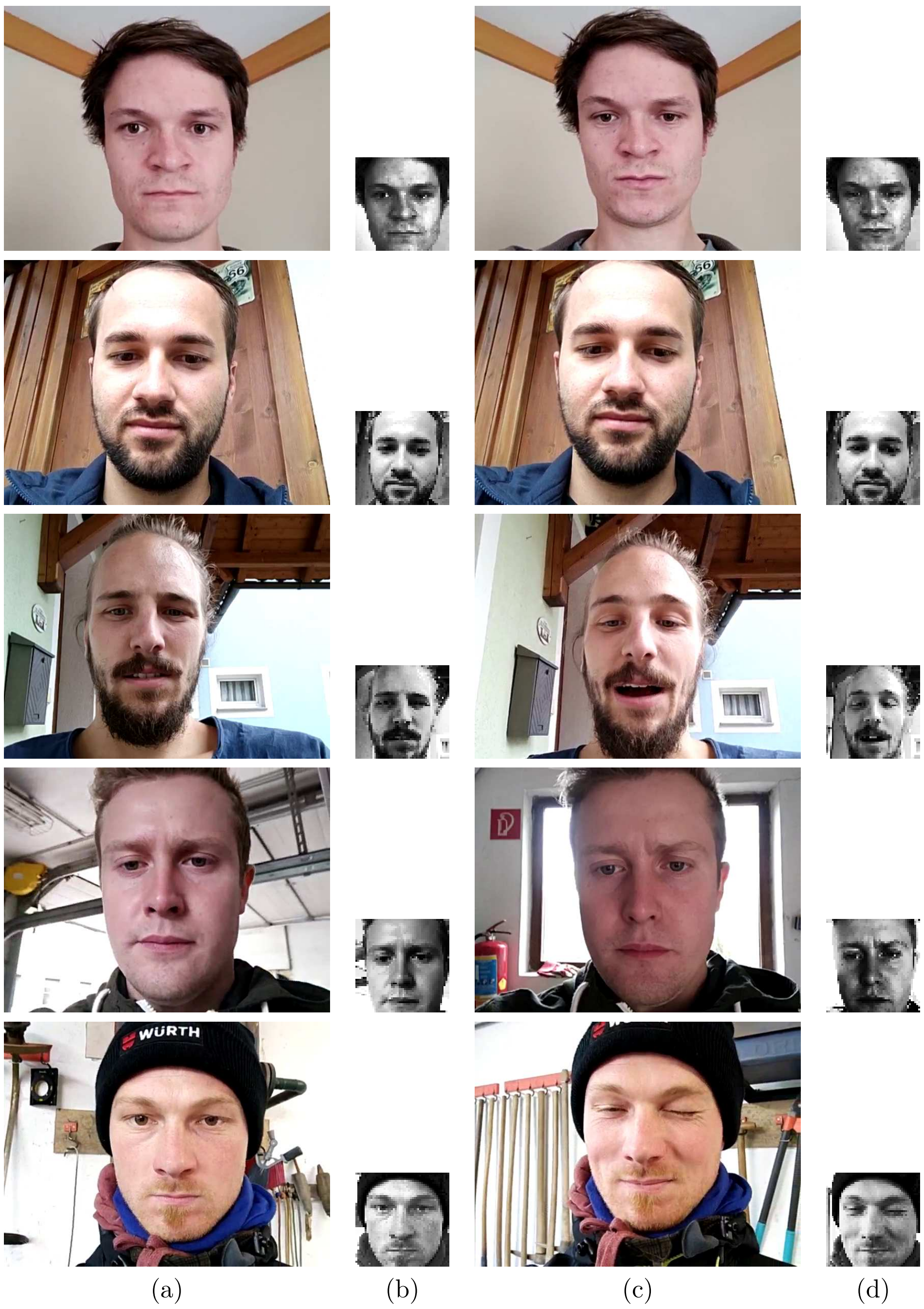


Figure 6.4: (a)(c) Image extracted from video sequence. (b)(d) Same example preprocessed.

6.5 Test Data for Model Evaluation

For testing the trained model some special video sequences were recorded in addition to the remaining 50% of the face-database. The participant videos contain only one person per video, the additional evaluation data however can contain multiple persons in a single video-sequence. Some of the evaluation video-sequences also contain periods where no persons are seen at all and only black frames are available. Those special video-sequences look like described in the following list:

- Owner only.
- Owner is followed by a pause, followed by the owner again.
- Owner is immediately followed by the allegedly attacker.
- Owner is followed by a pause, followed by the attacker afterward.
- Attacker only.
- Attacker is immediately followed by the owner.
- Attacker is followed by a pause, followed by the owner afterward.

Table 6.1 gives a detailed listing of the face-database which are used for evaluating the continuous mobile face authentication approach.

Participants	**Sum of Videos**	**Owner Videos**
Train Dataset		
10	29 (3 videos each)	9
Evaluation Dataset		
12	36 (3 videos each)	9

Table 6.1: Data partitioning for training and evaluating the classification model.

Chapter 7

Evaluation

This chapter describes the implementation and evaluation of the prototypes along the creation of the continuous mobile face authentication approach. The implementation of the finished prototype was a step by step process where the functionalities of the prototypes were evaluated and improved in every step of the process. The first step (see section 7.1) is a prototypical implementation of a face recognition system on a Windows desktop computer. This face recognition prototype already uses OpenCV's HAAR feature-based cascade classifier based on Viola and Jones' algorithm to detect faces in the images [52]. This approach however is not able to recognize people beyond the test database (test data which is used for training the classifier model). This problem was solved in the second step (see section 7.2). This prototype is able to transform a N-class problem into a two-class problem, thus the prototype is able to predict persons beyond the test database. This is possible with the implementation of the difference image module. Also a very important part in this step is the weight function module. This function weights the importance of the face observation sample depending on the elapsed time between two observations. To save the confidence value which is responsible for the owner and non-owner recognition in the face authentication approach a second module, a simplified hidden Markov model is used. The benefit of this module is that only the previous confidence score and the current face observation are needed to calculate a new confidence score. Therefor the history of previous observations is obsolete. Also the Android recording application was implemented and the consequent video face database was recorded in this step to train the classification model and evaluate the finished continuous mobile face authentication prototype. In the third and last step (see section 7.3) a time decay module was introduced. This is necessary to give the confidence value a decay over time when for a period no new face samples can be detected.

The final continuous mobile face authentication system was evaluated in form of a case study. One participant was predefined as the owner of the smartphone and the remaining 22 participants are all allegedly attackers. In the evaluation scenario, the owner of the smartphone is static and never changes. Further special scenarios also always have the same predefined participant as owner (this applies for section 7.2 and section 7.3).

7.1 Recognition Prototype

In this section we present our first approach towards continuous mobile face authentication. We implemented a face recognition prototype to identify different people within a small prerecorded video database. This video database included five videos of five different people each. Each video was labeled corresponding to the person for later identification. Images were recorded with a laptop's web camera and merged together to image-stacks. 80% of the image-stacks were used for training the recognition classifier model, the remaining 20% were used for model evaluation. This prototype is not able to predict on people beyond the face-database, which will be implemented in the next step of the implementation process. To create the classifier model all frames were extracted of the image-stack. Frames with a face in it were used for model training, the rest was not used and ignored. For the evaluation also only the frames with a face in it were used. This approach is just able to recognize participants within the database. Therefor the classifier model predicts five different probabilities for five different participants.

7.1.1 Recognition Prototype Toolchain

Figure 7.1 gives a short overview of the modules used in the recognition prototype and the process to come up with the probabilities for each participant. When the input image-stack is from an unknown person beyond

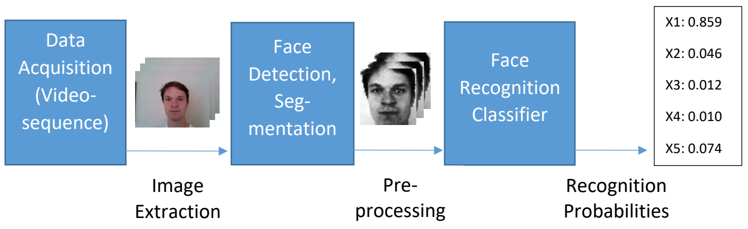

Data Acquisition (Video-sequence)
Face Detection, Seg-mentation
Face Recognition Classifier
X1: 0.859
X2: 0.046
X3: 0.012
X4: 0.010
X5: 0.074
Image Extraction
Pre-processing
Recognition Probabilities

Figure 7.1: Overview of modules used in the recognition prototype. At the end of the toolchain five different probabilities for five different persons are calculated.

the test video database this person will be identified as one person within the database which is most similar to the unknown one. The fact that this prototype is not able to predict on unknown people makes it inappropriate for our continuous mobile face authentication approach and therefor needs to be improved.

7.1.2 Results Recognition Prototype

The extracted images of the image-stacks which where used at this stage of the prototype are similar to the extracted images of the finished continuous mobile face authentication approach (see section 7.3). This prototype ignores images where no face is detected and therefore in every used frame a face is available and a prediction can be made. The following figure 7.2 shows the prediction on an image-stack containing images of ‘user1‘. The prediction in

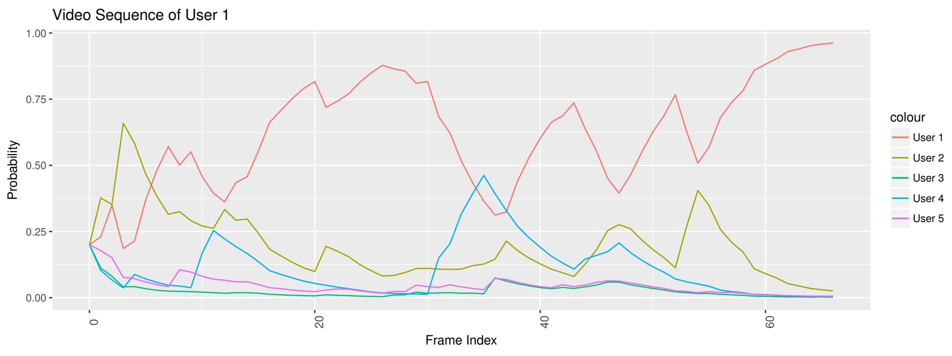


Figure 7.2: Face recognition on an image-stack containing images of ‘user1‘.

this image-stack is inaccurate and only on a short period in the middle and in the end recognizes the participant properly. Figure 7.3 shows the prediction on an image-stack of ‘user2‘. The middle section of the prediction graph is

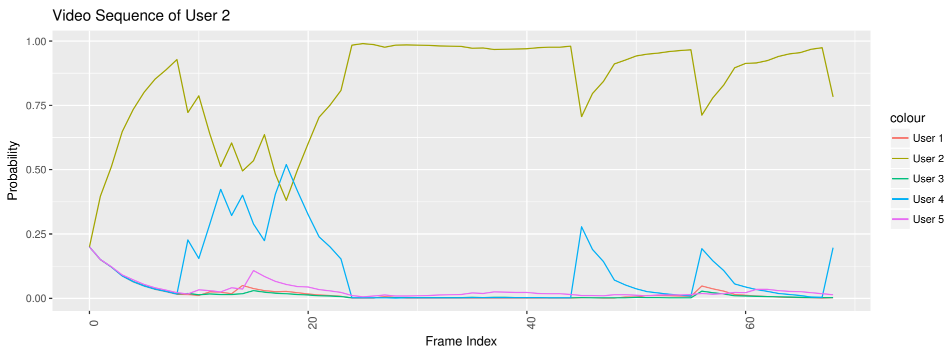


Figure 7.3: Face recognition on an image-stack containing images of ‘user2‘.

positive and nearly on 100% probability. The probability indicates that the image-stack belongs to the correct user. The start of the prediction though was similar to the prediction on ‘user1‘ pretty inaccurate and jittery. The last prediction (see figure 7.4) shows the prediction on the image-stack of ‘user3‘. This prediction is the best out of the given three examples. Only the

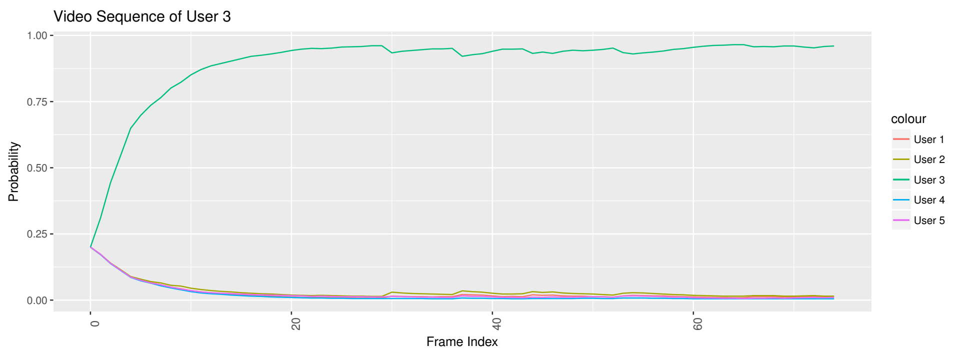


Figure 7.4: Face recognition on an image-stack containing images of ‘user3‘.

start gives bad prediction result for the given user. The rest of the prediction graph is positive and nearly on 100% probability.

7.1.3 Discussion

We assume that the inaccurate predictions are the result of the lack of data used at this stage of the prototype. We assume that we would achieve a better prediction rate on the different participants when more image-stacks would be used for training. This would lead to a better prediction accuracy which we evaluated in the later sections in this chapter (see section 7.2 and section 7.3). The different lengths of the prediction graphs depict the fact that only frames with a face in it are considered in this stage of the prototype. This is the reason for the unequal lengths among the evaluation image-stacks. The findings of the first prototype show that some improvements are necessary for a useful approach. Modules like the weight function need to be implemented as well as the transformation from the multi-class problem to a two-class problem to come up with a proper continuous mobile face authentication approach. But the basic detection module and the preprocessing of the images are useful and can be used in the improved prototype.

7.2 Authentication Prototype

In this section we describe the next step towards our continuous mobile face authentication prototype. The biggest improvement in this step is the prediction on video-sequences of unknown people (peolpe beyond the test face-database). The improvement which makes this possible is the implemented difference image creation module, which is not implemented in the previous approach. Another important step towards the finished prototype is the weight function module. Without this function every observation sample would be weighted equally and the elapsed time between two consecutive observations would be irrelevant. The time though is an important factor in continuous authentication systems in order to react in time when a switch between owner and non-owner happens. If an owner-frame is followed by a non-owner frame and the elapsed time between those two frames is high the last observation should be weighted more and should have more impact when calculating the new confidence value. Is the elapsed time between two frames high, the old confidence value (sum of previous observations) should be weighted lower than the new observation and vice versa. One part which is missing in this approach is the confidence decay over time, which was added in the last step of the implementation process. To get an overview of the toolchain of the prototype the used modules are depicted in figure 7.5.

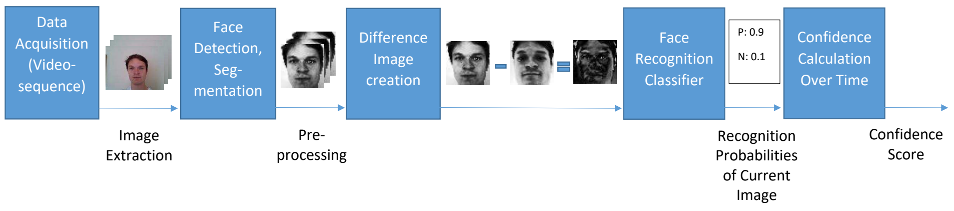

Data Acquisition (Video-sequence)
Image Extraction
Face Detection, Seg-mentation
Pre-processing
Difference Image creation
Face Recognition Classifier
P: 0.9
N: 0.1
Recognition Probabilities of Current Image
Confidence Calculation Over Time
Confidence Score

Figure 7.5: Overview of modules used in the continuous mobile face authentication system.

7.2.1 Difference Image Module

In order to transforms a multi-class problem $\{person_1, person_2, ..., person_N\}$ into a two-class problem $\{owner, non-owner\}$, which is necessary for our approach, the extracted and preprocessed images need to get transformed into difference-image space. This is done by calculating a difference image by subtracting one preprocessed image from another. In the model training stage if both images are of the owner (depending on the image label) the created difference image is labeled with ‘P‘ for positive sample, otherwise with ‘N‘ for negative sample. Those labeled samples are used for training. In the evaluation stage the observed sample will be transformed into a set of difference images. The set-size is depending on the size of the predefined reference image-set (35 reference images in our approach). Those reference images are used to transform a new observation image into 35 images in the difference-images-space. The images in difference-image space are used to calculate an average probability to determine if the new observation is from the owner or from another participant. This average probability is used in addition with the weight function and the old confidence value to calculate the new confidence value (the finished prototype also uses the decay function in addition, see section 7.3). Depending on the amount of those reference images a tradeoff between calculation speed and calculation accuracy need to be done. Using more reference images leads to a better accuracy but slows the calculation speed and vice versa. Some of the owner and non-owner difference-images are shown in figure 7.6. The darker the difference-images appear the more similar are the two input images.

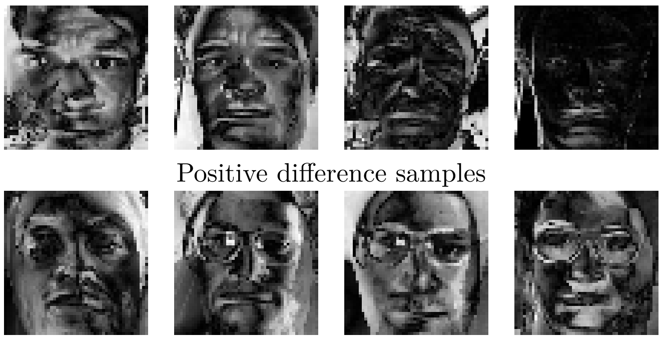
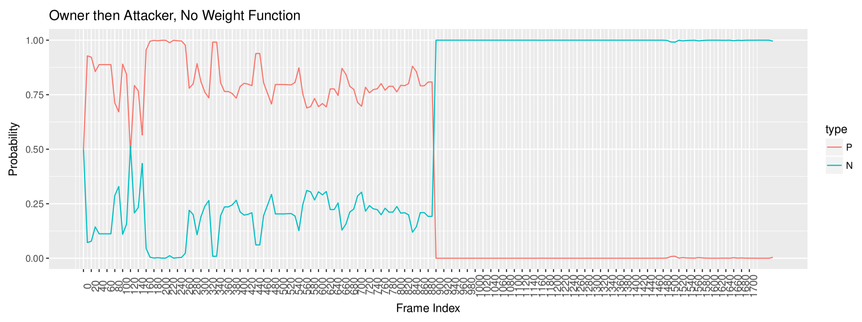


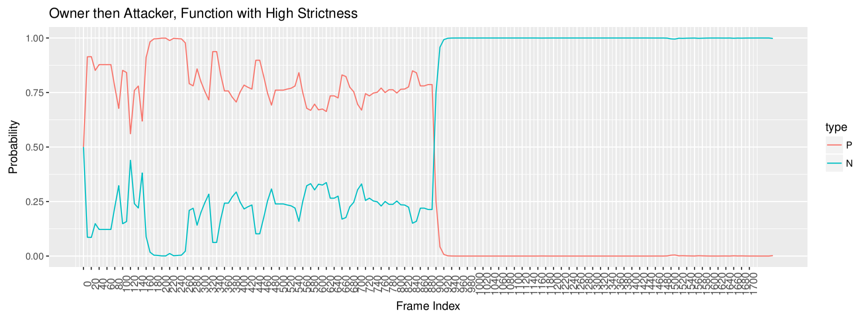
Figure 7.6: Difference images created out of two preprocessed gray-scale images. The darker the image appears the more similar are the input images.

7.2.2 Weight Function

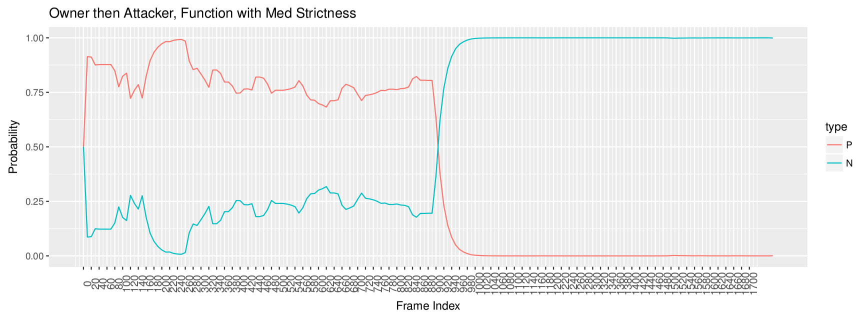
In this step of the implementation process we also used a weighting function to weight the importance of the observation sample based on the elapsed time to the previous observation. Depending on the strictness of the weight function a switch between two participants (owner, and non-owner) will be recognized sooner or later. We predefined a threshold of 0.7, which the positive value has to exceed in order to count as positive owner prediction. In the following images four different weight function constants and the resulting strictness is depicted (see figure 7.7 and figure 7.8). ‘Strictness‘ is the term which is used to describe the output value of the weight function and the consequent importance of the new observation sample. The images in figure 7.7 and figure 7.8 show the probability graph of a video-sequence where in the first half the owner and in the second half a non-owner is visible (switch on index 880). The different images in the figures are created with a different strictness of the weight function. Figure 7.7 and figure 7.8 distinguish in the allegedly attacker in the second half of the video. Videos of the participant from figure 7.8 are not used in the model training, however videos of the participant from figure 7.7 were used in the model training process (the videos for model training are different to the evaluation video-sequences though). We assume that this is the reason why the attacker in figure 7.7 is recognized that good. The strictness in the images ranges from no weight function in the first image, to a weight function with low strictness in the last image (high weight function constant). A weight function with a high strictness means in our case that the old confidence value has less impact compared to the new observation value, when calculating the new confidence score and vice versa. This is visible in the second image of figure 7.7 and figure 7.8 where the positive confidence graph is jittery and nearly the same as the confidence graph in the first image without any weighting function. A lower strictness results in a smoother confidence graph. New observations do not have that much impact on the new confidence value calculation compared to all other observation values before (the current confidence score is the result of all previous confidence score calculations), even if they are extreme high or extreme low (outliers). This is depicted in figure 7.8 between index 370 and 390 where one outlier (a false negative prediction, actual owner sample but predicted as non-owner) impacts the confidence graph more or less depending on the weight function strictness.



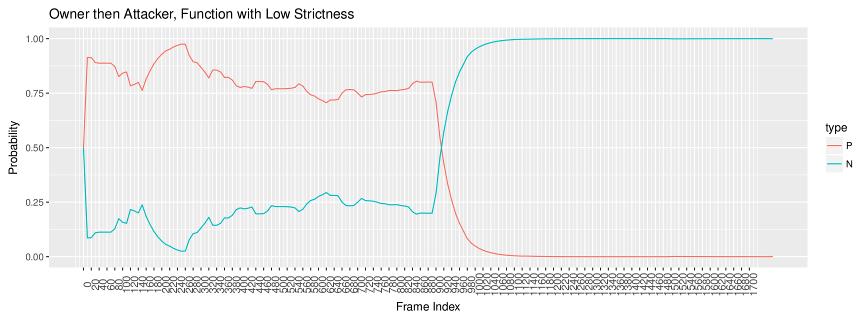
(a)



(b)

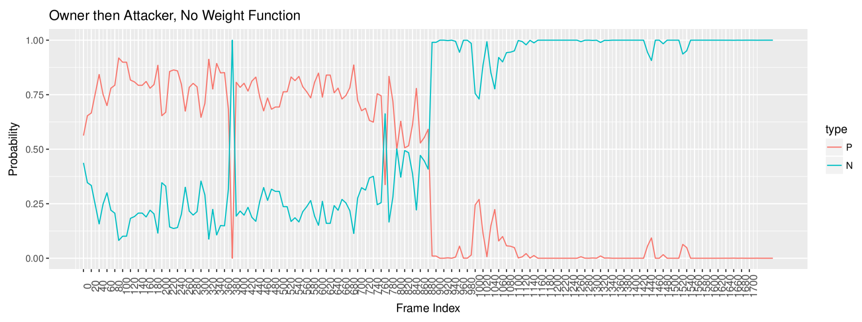


(c)

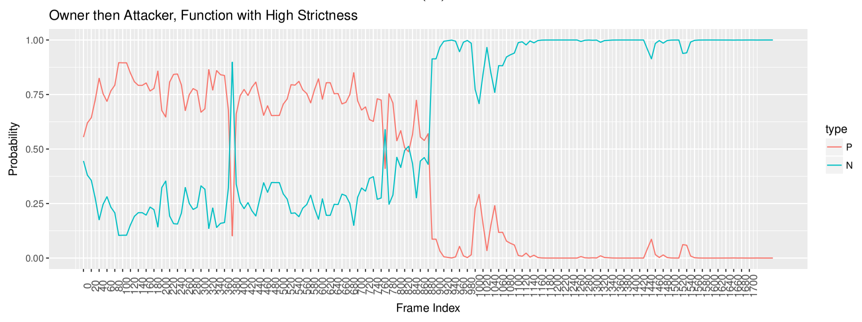


(d)

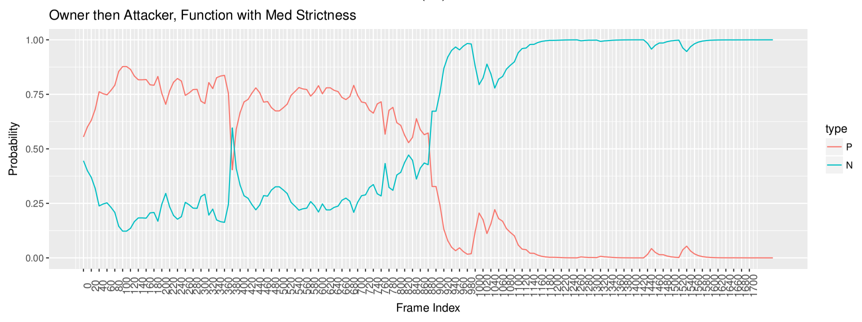
Figure 7.7: Predefined threshold of 0.7. In index 880 the switch between owner to non-owner happened. Depending on the strictness, and the corresponding weighting of new observations, the confidence value (positive) will increase faster or lower. (a) shows the graph without any weighting, (b) with high strictness weighting, (c) medium strictness weighting and (d) low strictness weighting.



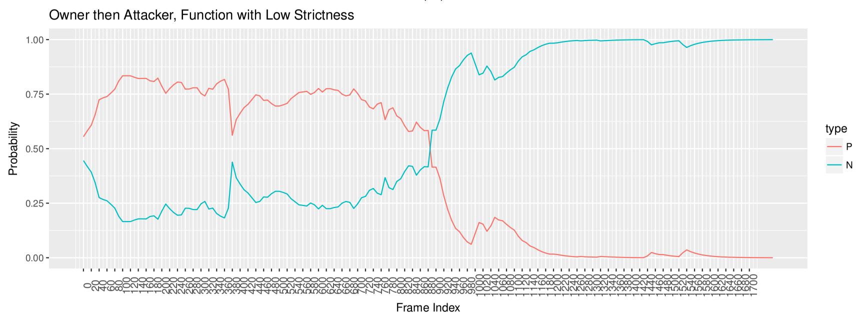
(a)



(b)



(c)



(d)

Figure 7.8: Predefined a threshold of 0.7. In index 880 the switch between owner to non-owner happened. Depending on the strictness, and the corresponding weighting of new observations, the confidence value (positive) will increase faster or lower. (a) shows the graph without any weighting, (b) with high strictness weighting, (c) medium strictness weighting and (d) low strictness weighting.

7.2.3 Test Setup

For model training and evaluating we used a data partition where approximately 50% of the face-database videos are used for training the classification model and the other 50% are used for evaluating the trained model. For this approximately the half of the face-database is randomly chosen and used to create the difference images for training the classification model. To get those difference samples every image needs to be subtracted of every other image of the database and labeled with P for owner samples and N for non-owner samples. This would lead to approximately 1.67 million difference image samples (Gaussian empirical formula). Therefore not every image is considered in the creation process and some are skipped. The other 50% of the face-database are used for evaluating the trained model with unknown data. The predefined owner however, has more than just three videos to get a good ratio between owner and non-owner samples of approximately 1:2 for training the model (for a detailed listing see table 6.1).

To calculate the difference images and predict on those, to recognize either the owner or the allegedly attacker, a set of reference images of the owner is necessary. Therefor a set of randomly chosen images (35 reference images in our approach) is extracted and used for this purpose. The extracted 35 images are not used for training the model classifier. In addition to the owner and attacker videos some special evaluation video-sequences (with owner and attacker) are recorded and used to test the switch between owner and non-owner and how long the system needs to react to this switch. The function which is responsible for this is the weight function.

7.2.4 Results Authentication Prototype

The following four boxplots in figure 7.9 show the distribution of the predicted P and N probabilities. To come up with the boxplots one frame per second is considered and the corresponding probabilities are used of this exact frame. In this case the P and N probabilities are both corresponding to the non-owner videos compared to the owner (negative class). The P and N probabilities sum up to 1 in each frame. A perfect boxplot result would only show predictions of the N probability over the threshold of 0.7. The first three boxplots show the probabilities of all participants for one light and ambient condition each. The last boxplot shows the probabilities of all participants for all light and ambient conditions combined (in the wild probabilities). The participants which are used for this evaluation are not used for training the model classifier and therefor should give a prospect of the system performance on unknown people. In this case the outdoor daytime condition (see (c) figure 7.9) has some false predictions. Reason for the overall false positive samples in the outdoor daytime condition video-sequences are the wrong predictions specially on two people (see figure 7.10).

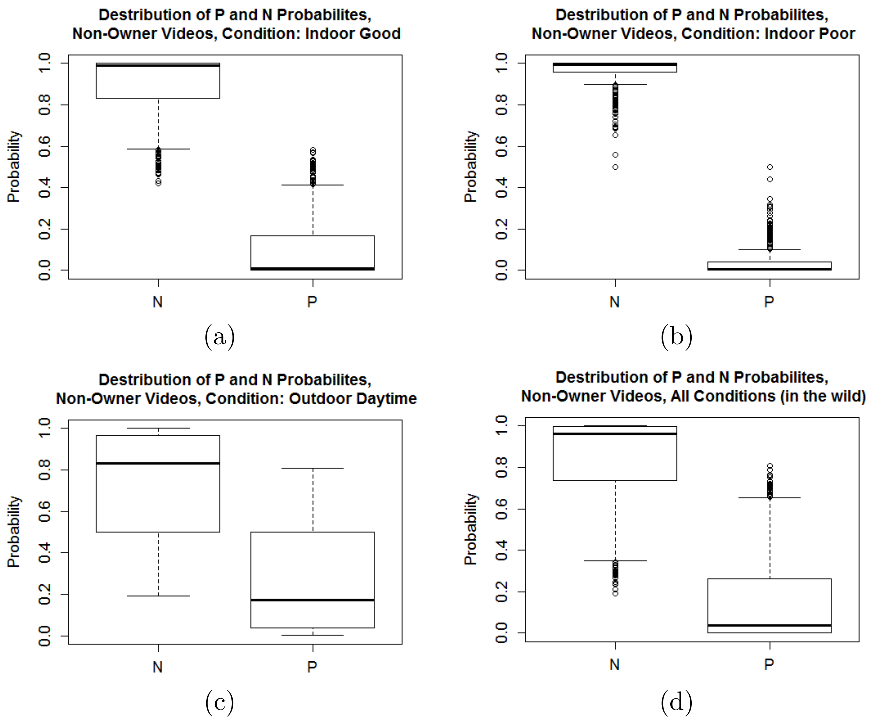


Figure 7.9: Distribution of the predicted P and N probabilities in non-owner video-sequences in different lighting and ambient conditions. (a)(b) and (c) each shows the distribution of one light condition at a time. (d) shows the distribution of all light conditions combined.

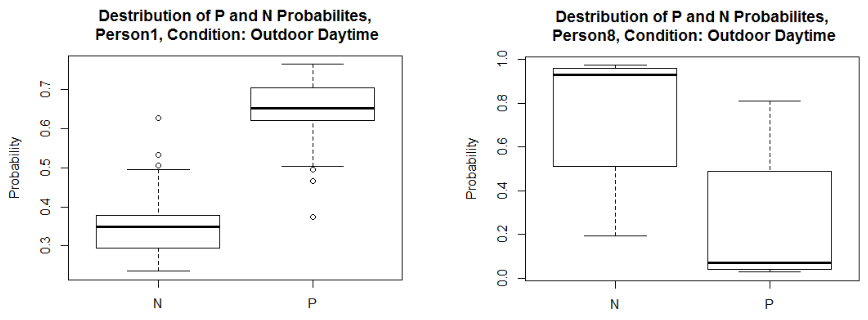


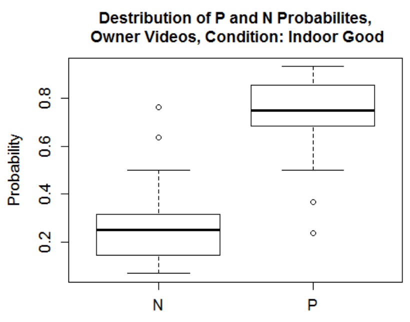
Figure 7.10: Distribution of the predicted P and N probabilities on one person each in condition outdoor daytime. Those two people have big impact on the overall prediction probability.

The next four boxplots in figure 7.11 are the same as the previous ones but now with P and N probabilities of one frame per second corresponding to the owner videos compared to the owner (positive class). The classifier model is trained with videos of the owner. However the videos for training the model and the videos for evaluating the model are different. The P and N probabilities again sum up to 1 in each frame. A perfect boxplot result would only show predictions of the P probability over the threshold of 0.7. The first three boxplots show the probabilities of all owner video for one light and ambient condition (three videos each). The last boxplot shows the probabilities of all lighting and ambient conditions combined (in the wild probabilities). In this case some false prediction outliers are specially in the indoor poor light condition (see (b) figure 7.11).

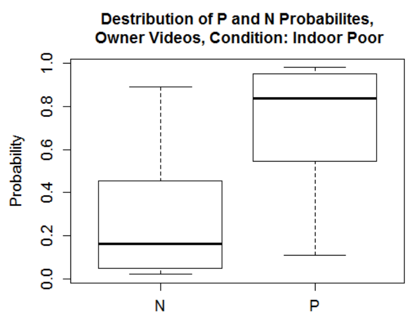
With the the P and N probabilities considering one frame per second of all video-comparisons (non-owner videos compared with owner and owner videos compared with owner) the true positive rates (TPR)and the false positive rates (FPR) can be calculated. A P prediction value over the threshold of 0.7 is considered as positive prediction and a N prediction value under the threshold of 0.7 is considered as negative. In table 7.1 the TPR and FPR are stated for every light condition, which indicate the performance of our approach.

Condition	**TPR**	**FPR**
Indoor Good	0.774	1.0
Indoor Poor	0.715	1.0
Outdoor Daytime	0.84	0.970
All (In-the-wild)	0.789	0.988

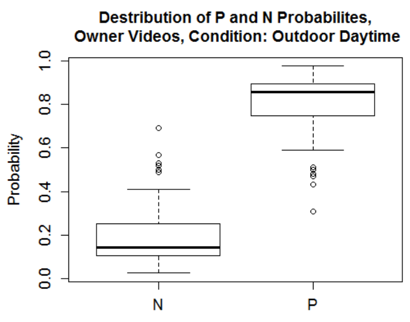
Table 7.1: true positive rates (TPR) and the false positive rates (FPR) calculated for each condition.



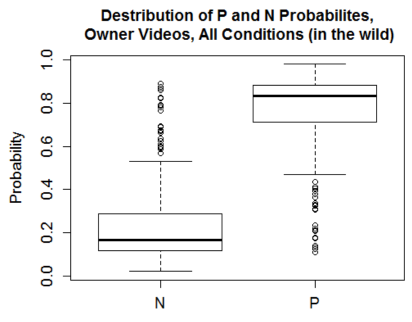
(a)



(b)



(c)



(d)

Figure 7.11: Distribution of the predicted P and N probabilities in non-owner video-sequences in different lighting and ambient conditions. (a)(b) and (c) each shows the distribution of one light condition at a time. (d) shows the distribution of all light conditions combined.

7.2.5 Discussion

The boxplots in figure 7.9 and figure 7.11 depict the fact that some conditions perform better for a specific class than others. Also differences between the conditions themselves are visible. In the non-owner videos the prediction density on the outdoor condition scatter more than in the other two conditions. Also the prediction on owner videos perform worse in the indoor poor lighting condition than the others, in terms of probability density scatter. We assume that we could improve the prediction rate on videos with stubborn conditions with increasing the number of training videos for those stubborn lighting and ambient conditions.

We decided to use the medium strictness for the weight function which is a good trade off between accuracy and a good weight function output that does not impact the overall confidence score when new values are observed (which a low strictness weight function does for example). Assume we have a value of 100% confidence it takes 3580 milliseconds to drop from 100% to 69.9% (below threshold of 70%) confidence when only considering the old confidence value, which is quite fast. The finished confidence score calculation however requires the old confidence value and the new observation value in combination with the weight function (see section 5.5). The strictness of the weight function can be adapted and tuned for the different users and scenarios, so it is not necessary that we found the perfect value for the parametrization.

7.3 Improved Authentication Prototype

In this section we present the finished continuous mobile face authentication prototype. The implementation of the decay function is the essential improvement in this step of the implementation process. This is necessary to give the confidence value a decay over time when no new faces get detected, which was not implemented in the previous approach. This is a security measure to avoid attacks where an allegedly attacker gets access just by hiding the front facing camera. This could happen when the owner uses the smartphone and the confidence value is high enough to get access, the attacker could circumvent the authentication mechanism just by hiding the camera and thereby stop the face image acquisition whereby the confidence score stays the same. To prevent this attack we introduced the decay function to decrease the confidence score over time when this scenario occurs. To get an overview of the toolchain of the finished prototype the used modules are depicted in figure 7.12. The toolchain is the same as in the previous approach only the decay function calculation was considered in the confidence score calculation which was missing in the previous approach.

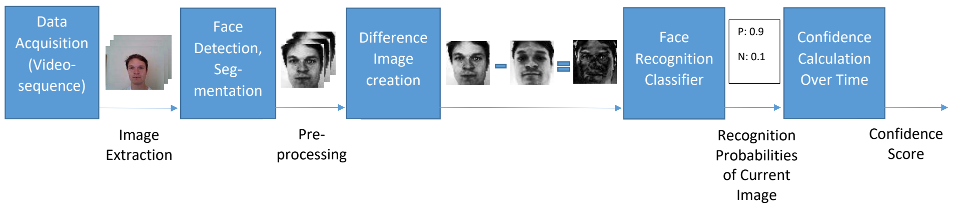
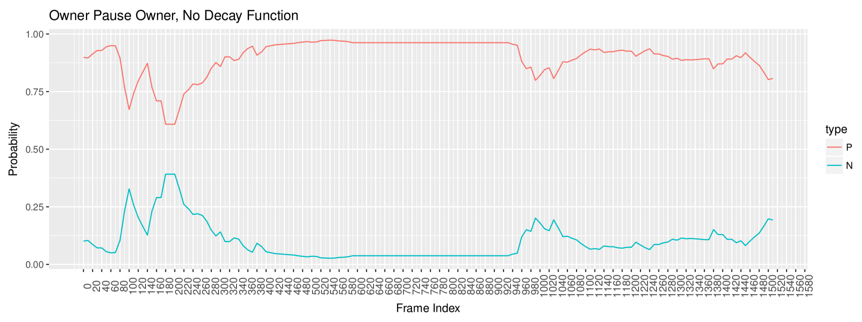

Data Acquisition (Video-sequence)
Image Extraction
Face Detection, Seg-mentation
Pre-processing
Difference Image creation
Face Recognition Classifier
P: 0.9
N: 0.1
Recognition Probabilities of Current Image
Confidence Calculation Over Time
Confidence Score

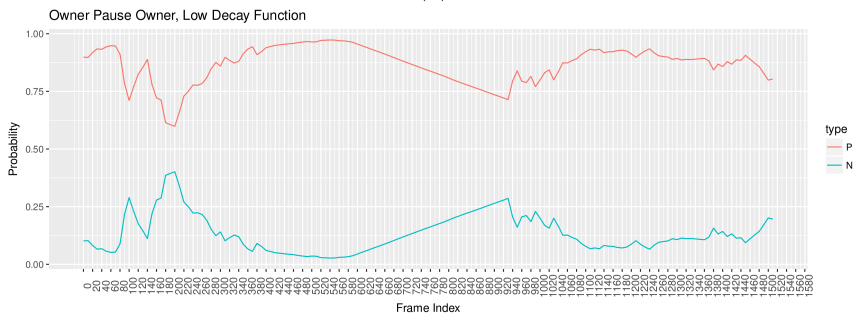
Figure 7.12: Overview of modules used in the continuous mobile face authentication system.

7.3.1 Decay Function

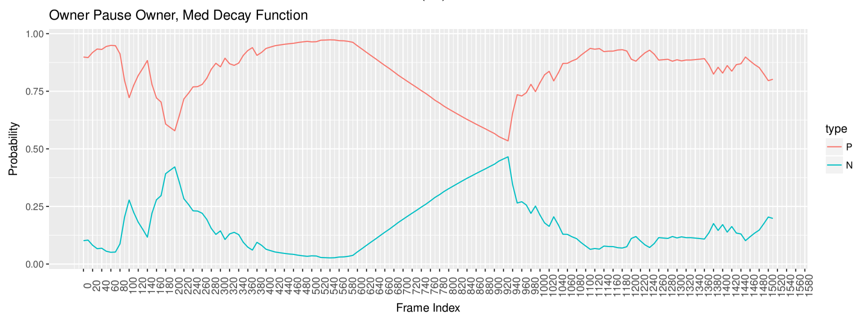
The last big implementation step was the consideration of time and the corresponding decay of the confidence score when no faces can get detected. This is important because without this function the confidence score would stay the same in the period where no new faces can get detected. Figure 7.13 depict different levels of decay. The images in figure 7.13 show the probability graph of a video-sequence where the owner is visible. The owner frames are interrupted by a period of plain black frames in the middle of the video-sequence (between index 570 and index 930). The first image shows the confidence graph without any decay. The confidence score stays at the same height during the period where no faces can get detected. The other three images show different intensities of the decay function ascending in intensity. The higher the intensity of the decay function the faster drops the positive confidence graph when no face can get detected. Index 570 in figure 7.13 is the first frame of the video-sequence without a face of the upcoming no-face-period. Depending on the decay intensity the faster or slower will the positive confidence value drop.



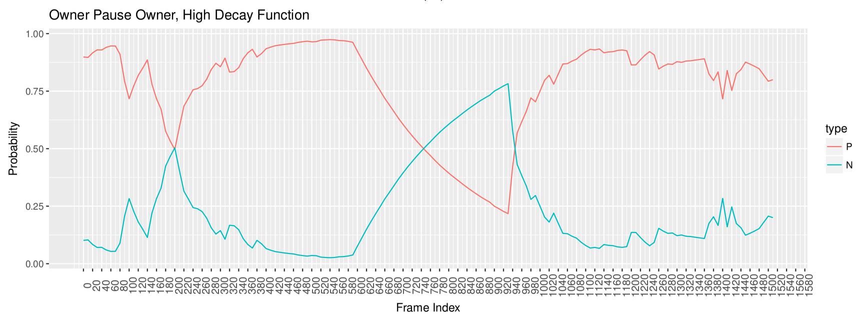
(a)



(b)



(c)



(d)

Figure 7.13: In index 570 the switch between face detection to non-face detection happened. Depending on the decay intensity the confidence value (positive) will decrease faster over time when no new faces can get detected. (a) shows the graph without any decay, (b) with low decay, (c) medium decay and (d) high decay.

7.3.2 Test Setup

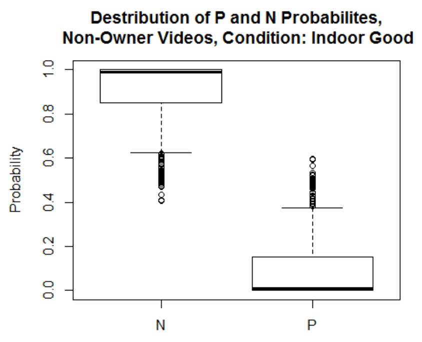
In this step of the prototype we again used for model training and evaluation a data partitioning of approximately 50% of the video-sequences for training the classification model and 50% for evaluating the trained model. We used the same half of the participants as in the evaluation of the authentication prototype in section 7.2. Also the predefined owner has again more videos than the other participants to get a good ratio between owner and non-owner samples for the model training (for a detailed listing see table 6.1). Also the owner reference images are the same as in the authentication prototype of section 7.2. For testing the decay function some special video-sequences were captured. Video-sequences with a short time where no faces can get detected (plain black frames) are recorded. This was acquired by hiding the front facing camera as described in the attack scenario.

For training the classification model we used the LibSVM model training methods with a set of parameters [10]. We used a linear SVM of type C_SVM (default classification type for the SVM) with a γ-parameter of 0.5 and a C-parameter of 1.

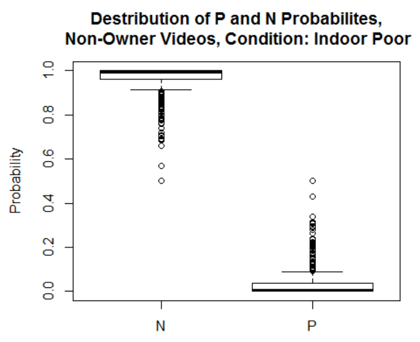
7.3.3 Results Improved Authentication Prototype

The following four boxplots in figure 7.14 show the distribution of the predicted P and N probabilities. To come up with the boxplots again one frame per second is considered and the corresponding probabilities are used of this exact frame. In this case the P and N probabilities are both corresponding to the non-owner videos compared to the owner (negative class). The P and N probabilities sum up to 1 in each frame. A perfect boxplot result would only show predictions of the N probability over the threshold of 0.7. The first three boxplots show the probabilities of all participants for one light and ambient condition each. The last boxplot shows the probabilities of all participants for all light and ambient conditions combined (in the wild probabilities). This is the same as in the evaluation in section 7.2.4 but with the additional decay function. The participants which are used for this evaluation are again not used for training the model classifier therefor these plots should give a prospect of the system performance on unknown people. In this case the outdoor daytime condition has some false predictions. The reason for the overall false positive samples in the outdoor daytime condition video-sequences are the wrong predictions specially on two persons (see figure 7.15). This is similar without the decay function in the previous prototype.

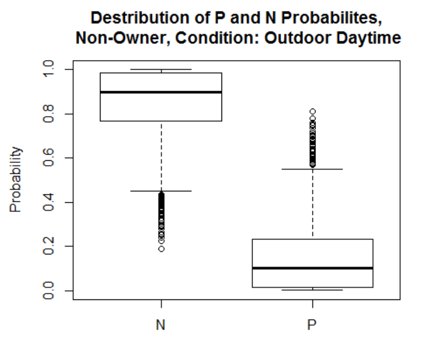
The next four boxplots in figure 7.16 are the same as the previous ones but now with P and N probabilities of one frame per second corresponding to the owner videos compared to the owner (positive class). The P and N probabilities again sum up to 1 in each frame. A perfect boxplot result



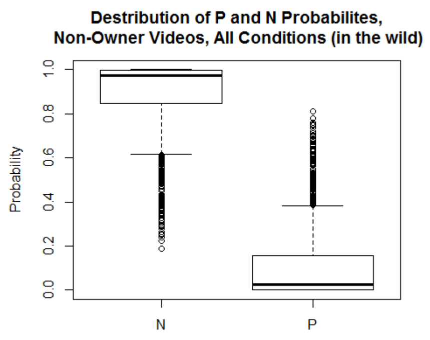
(a)



(b)

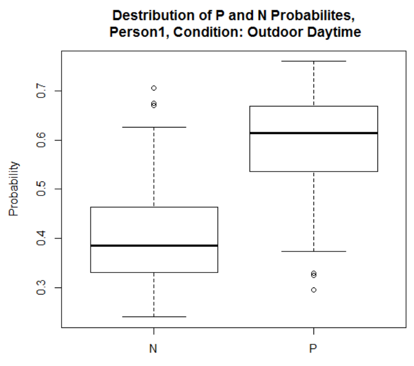


(c)



(d)

Figure 7.14: Distribution of the predicted P and N probabilities in non-owner video-sequences in different lighting and ambient conditions. (a)(b) and (c) each shows the distribution of one light condition at a time. (d) shows the distribution of all light conditions combined. The graphs show the results with decay functions.



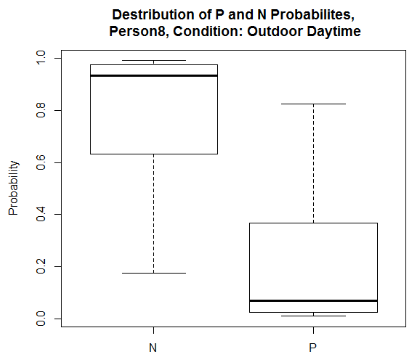


Figure 7.15: Distribution of the predicted P and N probabilities on one person each in condition outdoor daytime. Those two persons have big impact on the overall prediction probability.

would only show predictions of the P probability over the threshold of 0.7. In this case some false prediction outliers are specially in the indoor poor light condition (see (b) figure 7.16).

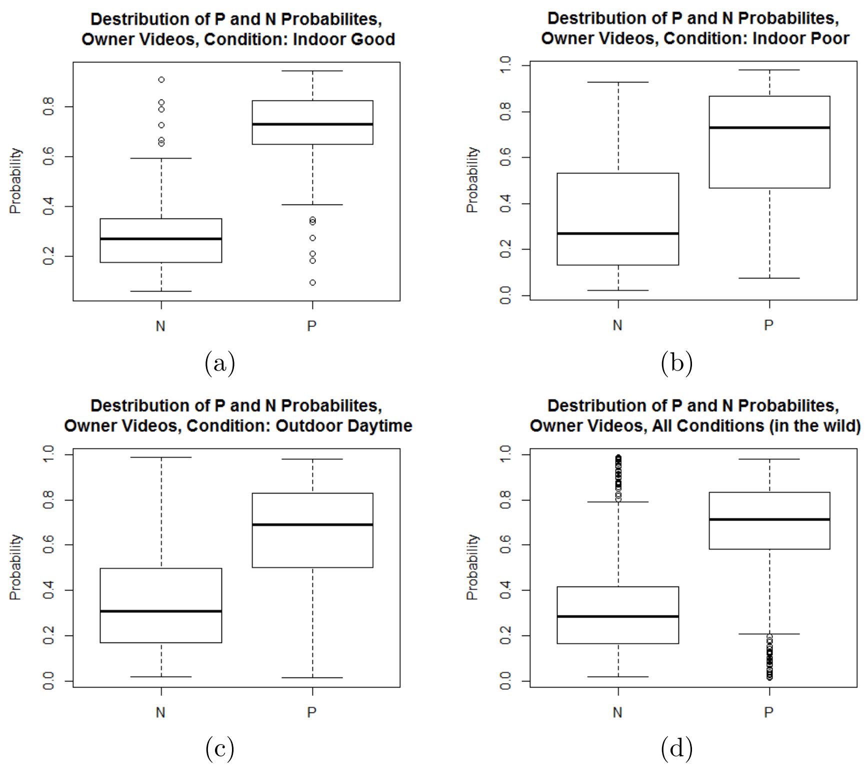


Figure 7.16: Distribution of the predicted P and N probabilities in owner video-sequences in different lighting and ambient conditions. (a)(b) and (c) each shows the distribution of one light condition at a time. (d) shows the distribution of all light conditions combined. The graphs show the results with decay functions.

With the the P and N probabilities considering one frame per second of all video-comparisons (non-owner videos compared with owner and owner videos compared with owner) the true positive rates (TPR) and the false positive rates (FPR) can be calculated. In table 7.2 the TPR and FPR are stated for every light condition with the decay function considered, which indicate the performance of our final approach. There are some false classifications though specially in the false negative samples (FN, actual owner and predicted as non-owner) which is depicted in the TPR values in table 7.2. Most of those false classifications are generated in the outdoor daytime condition which can be seen in the image (c) in figure 7.16. The overall prediction of 90.75% and the TPR and FPR results in table 7.2 though indicates a good prediction accuracy of the finished authentication prototype specially when predicting on allegedly attackers.

Condition	**TPR**	**FPR**
Indoor Good	0.582	1
Indoor Poor	0.539	1
Outdoor Daytime	0.483	0.980
All (In-the-wild)	0.528	0.992

Table 7.2: true positive rates (TPR) and the false positive rates (FPR) calculated for each condition with considered decay function.

The upcoming images in figure 7.17 show four different ROC-curves (receiver operating characteristic curve). A ROC-curve should give an overview of the performance of a binary ($\{owner, non - owner\}$) classifier at various threshold settings. Four different ROC-curves are depicted in figure 7.17. The first three graphs show the TPR versus FPR predictions of one of the conditions each, the last graph shows the prediction of all three conditions combined. A perfect ROC-curve would hug the upper left corner which means that the classifier does a good job separating the classes. A bad classifier would have a ROC-curve near or along the diagonal dashed line, which means that the classifier does nothing more than random guessing. In our case the classifier does a good job separating the classes. Only at some threshold settings will the negative and positive class prediction overlap which leads to a wrong prediction (see density-plot in figure 7.18).

7.3.4 Discussion

The box-plot images (see figure 7.14 and figure 7.16) show that the prototype with the decay function performs slightly better on non-owner video-sequences specially in the outdoor daytime condition in comparison to the prototype without the decay function. The owner videos have a higher negative probability percentage when the decay function is used than without decay. We assume that the reason therefor are the samples where no faces get detected and the positive confidence score drops on these samples. This lead to a higher negative percentage in the overall prediction which is depicted in the overall TPR of 0.582. An overall prediction accuracy of 90.75% and a FPR of 0.992 though are good indicator that our continuous mobile authentication approach performs good specially when detecting allegedly attackers. Also the density plot (see figure 7.18) and the ROC-curves (see figure 7.17) indicate a good performance of our approach with the predefined threshold.

We used a medium decay intensity which makes a good tradeoff between a fast reaction on no-face images (and the corresponding confidence score decrease) and a good function output that does not impact the overall con-

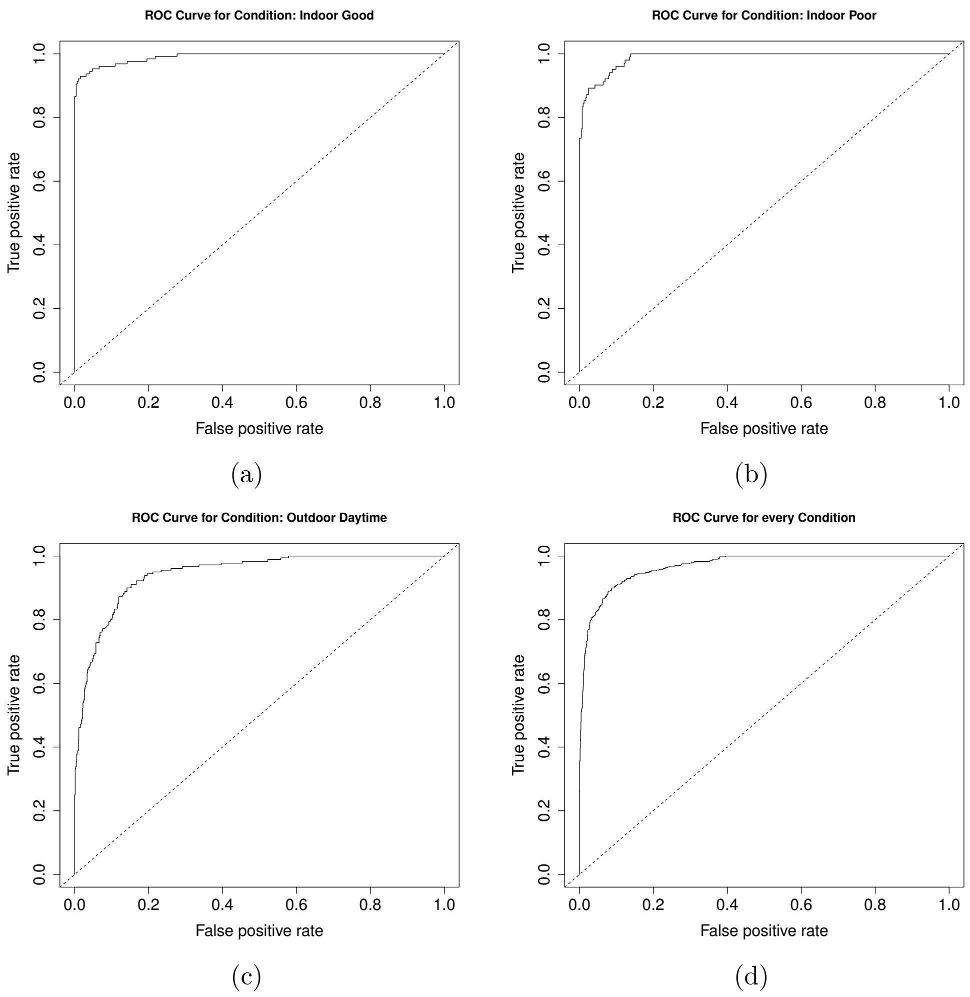


Figure 7.17: Comparison of the TPR and the FPR to depict the performance of the binary classifier.

fidence score when one outlier occurs (no-face image detected). If one single no-face detection would impact the overall confidence score, the graph would be jittery and drop below the threshold on some points only because of a single no-face detection. The intensity of the decay function can be adapted and tuned for the different users and scenarios, so it is not necessary that we found the perfect value for the parametrization.

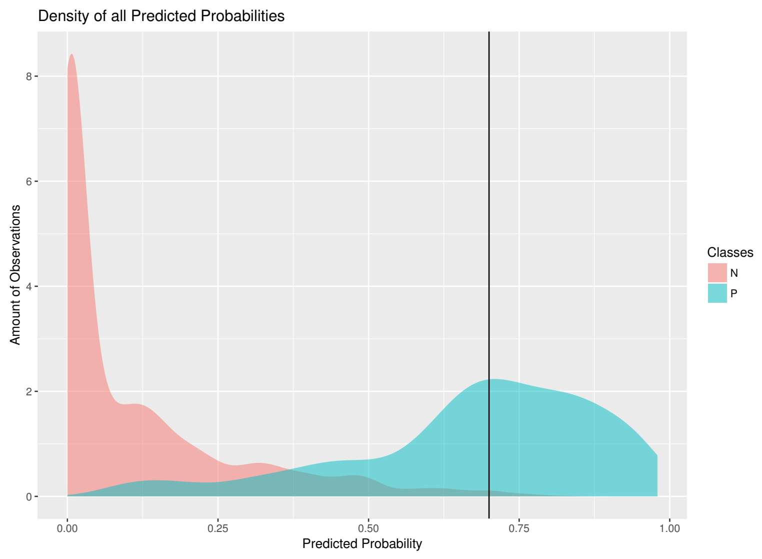


Figure 7.18: Density of the predicted probabilities of all videos of all participants. On the x-axis is the predicted probability and on the y-axes the amount of observations. The vertical line indicates our predefined threshold.

Chapter 8

Conclusion

We worked on a continuous mobile face authentication approach which is able to distinguish between the predefined owner of the smartphone and other users which might harm the owner by getting access to sensible data. There are plenty of other face authentication approaches on the market or currently in research, but most of them are explicit. There are some of the exceptions stated in the related work chapter 3. Explicit or active means that users have to explicitly look into the camera to be authenticated. The user experience is negatively affected by demanding the users attention on one hand and the users time on the other. In our approach users must not actively authenticate, they just have to work on the device, and automatically remain authenticated when they have the permission to. The continuous mobile face authentication system requires video-sequences or camera streams captured by the front facing camera and use them for authentication. In our approach we only consider the authentication of one smartphone owner against attackers, because a smartphone is usually privately owned and not shared by multiple users. The implementation of the finished prototype was a step by step process where the functionality of the prototypes were evaluated and improved in every step of the process.

Our first implementation was a prototypical implementation of a face recognition system which is able to recognize faces in images and distinguish between different persons. This face recognition prototype uses OpenCV's HAAR feature-based cascade classifier based on Viola and Jones' algorithm [52]. This prototype however is not able to detect people beyond the test database. Basic parts of the implementation in this step like the frame extraction from video-sequences, the preprocessing of those images and the face detection in those preprocessed images are working fine and provide the base of the finished prototype. The prototype is able to detect multiple faces in one image and only consider the biggest one when a minimum height and width is given. This should circumvent shoulder surfing and avoid detecting objects in the background of the user which are non-face objects.

The main improvements in the second step of the implementation process were the weight function and the simplified hidden Markov model to save the current state as well as the implementation of the difference creation module. The weight function is an important part of the toolchain which gives a new observation value an importance depending on the elapsed time between two consecutive observations. The evaluation shows that the introduction of the weight function works better in comparison to an approach without any weighting. With the weighting new observations do not have that much impact on the confidence value calculation compared to all other observation values before, even if the observation value is extreme high or extreme low (outliers). The difference creation module is able to transform a multi-class problem into a two-class problem which is necessary in our finished approach to predict on unknown persons. Therefor the module calculates the difference of two images by subtracting one image from another and label it as owner or non-owner image for the classifier training, or use it for prediction on unknown people. Also the Android recording application was implemented in this step so we were able to record the face-database. The face-database was used to train the face classification model and for evaluating different parts of the approach like predicting on unknown persons, testing the weight function and the decay function. A lesson we learned during the evaluation at an earlier stage in the implementation process was that a good data diversity in terms of lighting and ambient conditions makes big improvement in the performance, when predicting on unknown video-sequences. The data diversity was not good in the first prototypes and in the first tests, so we improved this with the recording of our own face-database. In terms of the results section though, we assume that there are some possible performance improvements by extending the face database and use more videos of more participants, specially in some conditions where the classifier do not perform as well. Therefore an extension of the face-database is the logical next step.

The last step towards the finished approach was the implementation of the decay function in order to give the confidence score a decay over time when for a period no new faces can get detected in the images. This is a security measure to avoid attacks where an allegedly attacker gets access just by hiding the front facing camera. Results indicate that the improved authentication approach with the decay function performs better when predicting on non-owner videos and a bit worse on owner video-sequences. We assume that the reason therefor are the samples where no faces get detected and the positive confidence score drops on these samples which is not the case in the prototype without any decay function. This lead to a higher negative percentage in the overall prediction TPR of 0.582, but with an overall prediction accuracy of 90.75% a FPR of 0.992 quite good though.

Our approach was evaluated in form of a case study because we predefined one participant as owner of the smartphone and the rest of the participants all as allegedly attackers. In the evaluation, the owner of the smartphone is static and never changes. An improvement for a future prototype and a more generic approach would be that any participant could be the owner of the device. This would require some improvements in the application in terms of a more generic classifier training phase. Another point left open by our current approach and implementation is the liveness detection of the user to avoid photo attacks. It is currently possible to trick the face detection module with photos of the smartphone owner to gain access. There is also room for improvement on preprocessing images in poor light conditions. In some of those images no faces can get detected because of the bad lighting which generates shadows in the participants faces and prevent the face detection module to detect the face. Challenges such as large diversity in lighting and ambient conditions which are present in the current mobile face authentication domain, also apply for our approach, and therefor might be in the focus of our future work too.

References

Literature

[1] Upal et al. "Partial Face Detection For Continuous Authentication". In: 2016 (cit. on pp. 9, 21, 23, 24).

[2] Abdulaziz Alzubaidi and Jugal Kalita. "Authentication of smartphone users using behavioral biometrics". *IEEE Communications Surveys & Tutorials* 18.3 (2016), pp. 1998–2026 (cit. on pp. 4, 6–8, 12, 13).

[3] Adam J Aviv et al. "Smudge Attacks on Smartphone Touch Screens." *Woot* 10 (2010), pp. 1–7 (cit. on pp. 1, 5).

[4] Hyeon Bae and Sungshin Kim. "Real-time face detection and recognition using hybrid-information extracted from face space and facial features". *Image and Vision Computing* 23.13 (2005), pp. 1181–1191 (cit. on pp. 9, 21–24, 34).

[5] Salil P Banerjee and Damon L Woodard. "Biometric authentication and identification using keystroke dynamics: A survey". *Journal of Pattern Recognition Research* 7.1 (2012), pp. 116–139 (cit. on p. 6).

[6] Ruud M Bolle et al. *Guide to biometrics*. Springer Science & Business Media, 2013 (cit. on p. 14).

[7] Frank Breitinger and Claudia Nickel. "User Survey on Phone Security and Usage." In: *BIOSIG*. 2010, pp. 139–144 (cit. on pp. 4, 8).

[8] William E Burr, Donna F Dodson, and William T Polk. *Electronic authentication guideline*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, 2004 (cit. on p. 3).

[9] Andrea Ceccarelli et al. "Continuous and transparent user identity verification for secure internet services". *IEEE transactions on dependable and secure computing* 12.3 (2015), pp. 270–283 (cit. on p. 26).

[10] Chih-Chung Chang and Chih-Jen Lin. "LIBSVM: A library for support vector machines". *ACM Transactions on Intelligent Systems and Technology* 2 (3 2011). Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 27:1–27:27 (cit. on pp. 36, 40, 68).

[11] Peter Corcoran and Claudia Costache. “Smartphones, Biometrics, and a Brave New World”. *IEEE Technology and Society Magazine* 35.3 (2016), pp. 59–66 (cit. on p. 11).

[12] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. *Machine learning* 20.3 (1995), pp. 273–297 (cit. on p. 31).

[13] Lorrie Faith Cranor and Simson Garfinkel. *Security and usability: designing secure systems that people can use.* " O’Reilly Media, Inc.", 2005 (cit. on p. 6).

[14] David Crouse et al. “Continuous authentication of mobile user: Fusion of face image and inertial measurement unit data”. In: *Biometrics (ICB), 2015 International Conference on.* IEEE. 2015, pp. 135–142 (cit. on pp. 21, 25, 42).

[15] Alexander De Luca and Janne Lindqvist. “Is secure and usable smartphone authentication asking too much?” *Computer* 48.5 (2015), pp. 64–68 (cit. on pp. 4–6).

[16] K Delac, M Grgic, and Tomislav Kos. “Sub-image homomorphic filtering technique for improving facial identification under difficult illumination conditions”. In: *International Conference on Systems, Signals and Image Processing.* Vol. 1. 2006, pp. 21–23 (cit. on p. 23).

[17] Mohammad Omar Derawi et al. “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.* IEEE. 2010, pp. 306–311 (cit. on pp. 9, 18).

[18] Rainhard Dieter Findling and Rene Mayrhofer. “Towards pan shot face unlock: Using biometric face information from different perspectives to unlock mobile devices”. *International Journal of Pervasive Computing and Communications* 9.3 (2013), pp. 190–208 (cit. on pp. 9, 20).

[19] George H Dunteman. *Principal components analysis.* 69. Sage, 1989 (cit. on p. 34).

[20] Serge Egelman et al. “Are you ready to lock?” In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security.* ACM. 2014, pp. 750–761 (cit. on p. 6).

[21] Mohammed E Fathy, Vishal M Patel, and Rama Chellappa. “Face-based active authentication on mobile devices”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on.* IEEE. 2015, pp. 1687–1691 (cit. on pp. 9, 21–23).

[22] Tao Feng et al. “Continuous mobile authentication using virtual key typing biometrics”. In: *Trust, security and privacy in computing and communications (TrustCom), 2013 12th IEEE international conference on.* IEEE. 2013, pp. 1547–1552 (cit. on pp. 16, 17).

[23] Dmitriy Fradkin and Ilya Muchnik. "Support vector machines for classification". *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 70 (2006), pp. 13–20 (cit. on p. 31).

[24] Mario Frank et al. "Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication". *IEEE transactions on information forensics and security* 8.1 (2013), pp. 136–148 (cit. on pp. 12, 16, 17).

[25] Hugo Gascon et al. "Continuous Authentication on Mobile Devices by Analysis of Typing Motion Behavior." In: *Sicherheit*. 2014, pp. 1–12 (cit. on p. 16).

[26] Zoubin Ghahramani. "An introduction to hidden Markov models and Bayesian networks". *International journal of pattern recognition and artificial intelligence* 15.01 (2001), pp. 9–42 (cit. on p. 33).

[27] Arnulf BA Graf, Alexander J Smola, and Silvio Borer. "Classification in a normalized feature space using support vector machines". *IEEE Transactions on Neural Networks* 14.3 (2003), pp. 597–605 (cit. on p. 23).

[28] Daniele Gunetti and Claudia Picardi. "Keystroke analysis of free text". *ACM Transactions on Information and System Security (TISSEC)* 8.3 (2005), pp. 312–347 (cit. on pp. 14, 15).

[29] Abdenour Hadid et al. "Face and eye detection for person authentication in mobile phones". In: *Distributed Smart Cameras, 2007. ICDSC'07. First ACM/IEEE International Conference on*. IEEE. 2007, pp. 101–108 (cit. on pp. 21, 24).

[30] Marian Harbach et al. "Its a hard lock life: A field study of smartphone (un) locking behavior and risk perception". In: *Symposium on usable privacy and security (SOUPS)*. 2014, pp. 9–11 (cit. on pp. 1, 5, 6).

[31] Marian Harbach et al. "Keep on Lockin'in the Free World: A Multi-National Comparison of Smartphone Locking". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM. 2016, pp. 4823–4827 (cit. on pp. 4, 6).

[32] Daniel Herrera, Juho Kannala, and Janne Heikkilä. "Joint depth and color camera calibration with distortion correction". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.10 (2012), pp. 2058–2064 (cit. on p. 21).

[33] Daniel Hintze et al. "A Large-Scale, Long-Term Analysis of Mobile Device Usage Characteristics". *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.2 (2017), p. 13 (cit. on p. 6).

[34] Thang Hoang, Deokjai Choi, and Thuc Nguyen. “Gait authentication on mobile phone using biometric cryptosystem and fuzzy commitment scheme”. *International Journal of Information Security* 14.6 (2015), pp. 549–560 (cit. on pp. 9, 18, 19).

[35] Thang Hoang et al. “A lightweight gait authentication on mobile phone regardless of installation error”. In: *IFIP International Information Security Conference*. Springer. 2013, pp. 83–101 (cit. on pp. 19, 20).

[36] Seung-eun Ji and Wooil Kim. “Whispered Speech Recognition for Mobile-Based Voice Authentication System” (2017) (cit. on p. 8).

[37] Amit Kale et al. “Gait-based recognition of humans using continuous HMMs”. In: *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*. IEEE. 2002, pp. 336–341 (cit. on pp. 18, 19).

[38] Christopher Kanan and Garrison W Cottrell. “Color-to-grayscale: does the method matter in image recognition?” *PloS one* 7.1 (2012), e29740 (cit. on pp. 22, 38).

[39] Samuel Karlin. *A first course in stochastic processes*. Academic press, 2014 (cit. on p. 33).

[40] Shri Karthikeyan et al. “Smartphone fingerprint authentication versus pins: A usability study (cmu-cylab-14-012)” (2014) (cit. on p. 8).

[41] Andrew J Klosterman and Gregory R Ganger. *Secure continuous biometric-enhanced authentication*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE, 2000 (cit. on p. 21).

[42] Arash Habibi Lashkari et al. “Shoulder surfing attack in graphical password authentication”. *arXiv preprint arXiv:0912.0951* (2009) (cit. on p. 5).

[43] Lingjun Li, Xinxin Zhao, and Guoliang Xue. “Unobservable Re-authentication for Smartphones.” In: *NDSS*. 2013 (cit. on pp. 16, 17).

[44] Xiaoming Liu and Tsuhan Cheng. “Video-based face recognition using adaptive hidden markov models”. In: *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2003, pp. I–I (cit. on p. 24).

[45] Ju Man and Bir Bhanu. “Individual recognition using gait energy image”. *IEEE transactions on pattern analysis and machine intelligence* 28.2 (2006), pp. 316–322 (cit. on pp. 9, 18, 19).

[46] Sanjit K Mitra and James F Kaiser. *Handbook for digital signal processing*. Book is not available as PDF. John Wiley & Sons, Inc., 1993 (cit. on p. 23).

[47] John V Monaco et al. “Developing a keystroke biometric system for continual authentication of computer users”. In: *Intelligence and Security Informatics Conference (EISIC), 2012 European*. IEEE. 2012, pp. 210–216 (cit. on pp. 14, 15).

[48] Kazuki Nakajima et al. “Footprint-based personal recognition”. *IEEE Transactions on Biomedical Engineering* 47.11 (2000), pp. 1534–1537 (cit. on p. 9).

[49] Claudia Nickel and Christoph Busch. “Classifying accelerometer data via hidden markov models to authenticate people by the way they walk”. *IEEE Aerospace and Electronic Systems Magazine* 28.10 (2013), pp. 29–35 (cit. on p. 19).

[50] Claudia Nickel et al. “Scenario test of accelerometer-based biometric gait recognition”. In: *Security and Communication Networks (IWSCN), 2011 Third International Workshop on*. IEEE. 2011, pp. 15–21 (cit. on p. 20).

[51] Mark S Nixon et al. “Automatic gait recognition” (1999) (cit. on pp. 9, 18).

[52] OpenCV. *OpenCV Open Source Computer Vision Library*. https://github.com/opencv/opencv, http://opencv.org/. 2017 (cit. on pp. 36, 37, 52, 74).

[53] Keyurkumar Patel, Hu Han, and Anil K Jain. “Secure face unlock: Spoof detection on smartphones”. *IEEE Transactions on Information Forensics and Security* 11.10 (2016), pp. 2268–2283 (cit. on p. 10).

[54] Keyurkumar Patel et al. “Live face video vs. spoof face video: Use of moiré patterns to detect replay video attacks”. In: *Biometrics (ICB), 2015 International Conference on*. IEEE. 2015, pp. 98–105 (cit. on pp. 10, 11).

[55] Vishal M Patel et al. “Continuous user authentication on mobile devices: Recent progress and remaining challenges”. *IEEE Signal Processing Magazine* 33.4 (2016), pp. 49–61 (cit. on pp. 9, 10, 20).

[56] Greig Paul and James Irvine. “IEDs on the Road to Fingerprint Authentication: Biometrics have vulnerabilities that PINs and passwords don’t”. *IEEE Consumer Electronics Magazine* 5.2 (2016), pp. 79–86 (cit. on pp. 8, 10).

[57] L. E. Peterson. “K-nearest neighbor”. *Scholarpedia* 4.2 (2009). revision #136646, p. 1883 (cit. on pp. 32, 33).

[58] Lawrence Rabiner and B. Juang. “An introduction to hidden Markov models”. *ieee assp magazine* 3.1 (1986), pp. 4–16 (cit. on p. 33).

[59] Shantanu Rane et al. “Secure biometrics: concepts, authentication architectures, and challenges”. *IEEE Signal Processing Magazine* 30.5 (2013), pp. 51–64 (cit. on p. 10).

[60] Pouya Samangouei, Vishal M Patel, and Rama Chellappa. “Facial attributes for active authentication on mobile devices”. *Image and Vision Computing* 58 (2017), pp. 181–192 (cit. on p. 24).

[61] Frode Eika Sandnes and Xiaoli Zhang. “User identification based on touch dynamics”. In: *Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC), 2012 9th International Conference on.* IEEE. 2012, pp. 256–263 (cit. on p. 16).

[62] Enrico Schiavone, Andrea Ceccarelli, and Andrea Bondavalli. “Continuous user identity verification for trusted operators in control rooms”. In: *International Conference on Algorithms and Architectures for Parallel Processing.* Springer. 2015, pp. 187–200 (cit. on p. 26).

[63] Mauricio Segundo et al. “Continuous 3d face authentication using rgb-d cameras”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.* 2013, pp. 64–69 (cit. on pp. 21, 24).

[64] Ali Sharifara, Mohd Shafry Mohd Rahim, and Yasaman Anisi. “A general review of human face detection including a study of neural networks and Haar feature-based cascade classifier in face detection”. In: *Biometrics and Security Technologies (ISBAST), 2014 International Symposium on.* IEEE. 2014, pp. 73–78 (cit. on pp. 29, 30).

[65] Terence Sim et al. “Continuous verification using multimodal biometrics”. *IEEE transactions on pattern analysis and machine intelligence* 29.4 (2007), pp. 687–700 (cit. on pp. 24, 25, 42, 43).

[66] Qian Tao and Raymond NJ Veldhuis. “Biometric authentication for a mobile personal device”. In: *Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on.* IEEE. 2006, pp. 1–3 (cit. on pp. 9, 20).

[67] IDonna Tapellini. *Smart phone thefts rose to 3.1 million in 2013.* http://www.consumerreports.org/cro/news/2014/04/smart-phone-thefts-rose-to-3-1-million-last-year/index.htm. [Online; accessed 11-May-2017]. 2014 (cit. on p. 4).

[68] Charles C Tappert et al. “A keystroke biometric system for long-text input”. In: *Optimizing Information Security and Advancing Privacy Assurance: New Technologies.* IGI Global, 2012, pp. 32–57 (cit. on pp. 14, 15).

[69] Shari Trewin et al. "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*. ACM. 2012, pp. 159–168 (cit. on pp. 6, 7, 10).

[70] Pei-Wei Tsai et al. "Interactive artificial bee colony supported passive continuous authentication system". *IEEE Systems Journal* 8.2 (2014), pp. 395–405 (cit. on pp. 21, 24).

[71] Matthew Turk and Alex Pentland. "Eigenfaces for recognition". *Journal of cognitive neuroscience* 3.1 (1991), pp. 71–86 (cit. on pp. 24, 33, 34).

[72] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2001, pp. I–I (cit. on pp. 21, 30).

[73] Paul Viola and Michael J Jones. "Robust real-time face detection". *International journal of computer vision* 57.2 (2004), pp. 137–154 (cit. on p. 29).

[74] Emanuel Von Zezschwitz, Paul Dunphy, and Alexander De Luca. "Patterns in the wild: a field study of the usability of pattern and pin-based authentication on mobile devices". In: *Proceedings of the 15th international conference on Human-computer interaction with mobile devices and services*. ACM. 2013, pp. 261–270 (cit. on p. 6).

[75] Phillip Ian Wilson and John Fernandez. "Facial feature detection using Haar classifiers". *Journal of Computing Sciences in Colleges* 21.4 (2006), pp. 127–133 (cit. on p. 29).

[76] Heiko Witte and Claudia Nickel. "Modular Biometric Authentication Service System (MBASSy)." In: *BIOSIG*. 2010, pp. 115–120 (cit. on p. 20).

[77] Jain-Shing Wu et al. "Smartphone continuous authentication based on keystroke and gesture profiling". In: *Security Technology (ICCST), 2015 International Carnahan Conference on*. IEEE. 2015, pp. 191–197 (cit. on p. 17).

[78] Pei-Yuan Wu et al. "Cost-effective kernel ridge regression implementation for keystroke-based active authentication system". *IEEE transactions on cybernetics* (2016) (cit. on pp. 14, 15).

[79] Qinghan Xiao and Xue-Dong Yang. "Facial recognition in uncontrolled conditions for information security". *EURASIP Journal on Advances in Signal Processing* 2010.1 (2010), p. 345743 (cit. on pp. 21–23).

[80] Ming-Hsuan Yang, David J Kriegman, and Narendra Ahuja. "Detecting faces in images: A survey". *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.1 (2002), pp. 34–58 (cit. on p. 28).

[81] Stefanos Zafeiriou, Cha Zhang, and Zhengyou Zhang. “A survey on face detection in the wild: past, present and future”. *Computer Vision and Image Understanding* 138 (2015), pp. 1–24 (cit. on p. 28).

[82] Cha Zhang and Zhengyou Zhang. *A survey of recent advances in face detection.* Tech. rep. Tech. rep., Microsoft Research, 2010 (cit. on p. 28).

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —

Breite = 100 mm
Höhe = 50 mm

— Diese Seite nach dem Druck entfernen! —