SACJ 28(2) December 2016
Research Article

User authentication based on continuous touch biometrics

Christina J. Kroeze[a], Katherine M. Malan[b, a]

[a] Department of Computer Science, University of Pretoria, Lynnwood Road, Pretoria, South Africa
[b] Department of Decision Sciences, University of South Africa, Hazelwood Campus, Pretoria, South Africa

ABSTRACT
Mobile devices such as smartphones have until now been protected by traditional authentication methods, including passwords or pattern locks. These authentication mechanisms are difficult to remember and are often disabled, leaving the device vulnerable if stolen. This paper investigates the possibility of unobtrusive, continuous authentication for smartphones based on biometric data collected using a touchscreen. The possibility of authenticating users on a smartphone was evaluated by conducting an experiment simulating real-world touch interaction. Touch data was collected from 30 participants during normal phone use. The touch features were analysed in terms of the information provided for authentication. It was found that features such as finger pressure, location of touch interaction and shape of the finger were important discriminators for authentication. The touch data was also analysed using two classification algorithms to measure the authentication accuracy. The results show that touch data is sufficiently distinct between users to be used in authentication without disrupting normal touch interaction. It is also shown that the raw touch data was more effective in authentication than the aggregated gesture data.

Keywords: continuous authentication, touch biometrics, verification

Categories: • **Security and privacy** ∼ **Biometrics** • Security and privacy ∼ Usability in security and privacy

Email:
Christina J. Kroeze christienkroeze@gmail.com,
Katherine M. Malan malankm@unisa.ac.za (CORRESPONDING)

Article history:
Received: 26 February 2016
Accepted: 21 October 2016
Available online: 9 December 2016

1 INTRODUCTION

Smartphones are extremely convenient devices, since they are so portable. Losing a smartphone can result in more damage than just the financial loss of replacing the device, because it may contain the owner's private information. Unfortunately, many smartphone users do not rate security as a high priority, and may disable their authentication mechanisms, leaving their devices vulnerable to attackers.

The purpose of a security mechanism is to hinder unauthorised users from gaining access to systems, but this could also interfere with authorised users' work. This leads to a tradeoff between security and usability in computer systems (Balfanz, Durfee, Smetters, & Grinter, 2004; Camp, 2007;

Kroeze, C.J. and Malan, K.M. (2016). User authentication based on continuous touch biometrics. *South African Computer Journal 28*(2), 1–24. https://doi.org/10.18489/sacj.v28i2.374

SACJ is a publication of the South African Institute of Computer Scientists and Information Technologists. ISSN 1015-7999 (print) ISSN 2313-7835 (online).

Cranor & Garfinkel, 2004; Sasse, Brostoff, & Weirich, 2001; Smetters & Grinter, 2002; Whitten & Tygar, 1999; Adams & Sasse, 1999; Yee, 2004). Authentication is at the forefront of the security vs. usability problem. Passwords have been used most popularly in authentication, since they are easy and cheap to implement (Bonneau, Herley, van Oorschot, & Stajano, 2012). Despite their popularity, passwords are generally considered to have both low usability and low security, because they rely on the user to remember them and to make them difficult to guess (Yan, Blackwell, Anderson, & Grant, 2004; Bonneau et al., 2012; Bonneau, 2010; Dhamija & Dusseault, 2008; Adams & Sasse, 1999; Florencio & Herley, 2007).

Biometrics are unique physiological or behavioural characteristics that are automatically measurable and distinct between individuals (Woodward, Orlans, & Higgins, 2003) and are becoming a popular alternative to passwords for authentication. Biometrics traditionally include fingerprints, voice or iris patterns and facial structure. What makes biometrics easier to use than passwords is that they do not place extra cognitive load on users and they do not rely on users' memory for authentication. Therefore, biometrics have a better chance of being both usable and secure and they do not rely on the user acting securely.

Biometrics are well suited to *continuous* (or unobtrusive) authentication. Continuous authentication is usually associated with behavioural biometrics such as keystroke dynamics, which can be measured without user intervention (Vildjiounaite, Mäkelä, Lindholm, Kyllönen, & Ailisto, 2007; Damousis, Tzovaras, & Bekiaris, 2008). Continuous authentication detects anomalies in user behaviour and can therefore secure a system at all times against imposters. It measures behavioural biometrics unobtrusively while the user continues to work as normal. This makes the authentication mechanism more usable and more secure as well, since the user is not interrupted and the system is continuously secured.

This paper proposes the use of touch biometrics for continuous, unobtrusive authentication on a mobile device such as a smartphone. Touch data was collected from 30 participants and analysed to determine whether the data could be used to authenticate users. Both raw touch data and aggregated gesture data were analysed in order to determine whether process intensive feature extraction rules are necessary to achieve good accuracy.

This paper is organised as follows. Section 2 discusses related work in biometric authentication on smartphones. Section 3 describes the processes followed to conduct the experiment. Section 4 analyses the classification accuracy of the collected touch biometric data. Lastly, Sections 5 and 6 present concluding remarks and suggest areas for future work.

2 RELATED WORK

Many studies have successfully tested biometric authentication on mobile phones. Some have focused on adding external sensors to the device such as a fingerprint scanner (NTT, 2003; Feng et al., 2012). Others have focused on built-in hardware for the detection of various biometric features (Derawi & Bours, 2013; Ho, Eswaran, Ng, & Leow, 2012; Agrawal, 2013; Kuseler, Lami, & Al-Assam, 2013). For example, the work of Derawi, Nickel, Bours, and Busch (2010), Derawi and Bours (2013) and Ho et al. (2012) focused on detecting changes in how a person walks. On Android version 4.0 a

facial recognition login tool was introduced. While it is a very user friendly way of authenticating, it is not very secure and can even be fooled with a static image of the phone's user displayed on another phone (Silverman, 2012).

In touch biometrics specifically, work has focused on augmenting existing smartphone authentication systems such as PINs, passwords and pattern locks with touch biometrics (Sae-Bae, Ahmed, Isbister, & Memon, 2012; De Luca, Hang, Brudy, Lindner, & Hussmann, 2012; Angulo, 2012; Saevanee & Bhatarakosol, 2008; Zheng, Bai, Huang, & Wang, 2012; Aviv, Sapp, Blaze, & Smith, 2012; Shahzad, Liu, & Samuel, 2013). That is, the authentication system still relies on the user remembering a PIN, password or pattern, but uses biometrics to validate that it is still the user inputting it. For example, a user enters a pattern to unlock the screen, and their touch data is analysed to determine if they are still performing the pattern in the same way as they did when they enrolled. However, this type of system depends on the user actively authenticating. It also relies on a limited set of actions that may be performed by each user, thereby increasing the accuracy of the system, but decreasing its flexibility.

Recent work has investigated *continuous* authentication using touch biometrics on smartphones (M. Frank, Biedert, Ma, Martinovic, & Song, 2013; Li, Zhao, & Xue, 2013; Feng et al., 2012). This type of authentication is not performed in an explicit, user initiated step. It is done in the background while a user is doing other tasks on the device. The following sections describe the approaches that have been used to investigate continuous touch biometrics.

2.1 Collection of touch data

One problem that arises in the investigation of touch biometrics is the manner of collecting such sensitive information from the operating system. Collecting touch data from an application is blocked by the operating system (OS) since it could lead to security problems.

Previous work has found various workarounds to this problem. The study by M. Frank et al. (2013) used a "mimic" interface to simulate the act of using the smartphone normally. Their system achieved low error rates. However, only two actions were investigated: swiping vertically and horizontally. Vertical swipes were recorded by asking users to read one of three randomly selected documents. Horizontal swipes were recorded by asking users to compare two images and switch between images with a horizontal swipe action. Their system therefore did not represent the complete set of real-world continuous interactions.

Li et al. (2013) used the device logs on the Android system to collect touch biometric data. These low level logs store data directly from the touch input device, if the user has root privileges on the OS.

Feng et al. (2012) augmented the smartphone's sensors with a digital sensor glove, recording hand movements in addition to touch data. Each participant performed three distinct gestures: swiping, pinching and spreading, and dragging or drawing shapes. These gestures were kept separately in the database. The use of distinct gestures indicates that this system was tested on a limited range of touch data.

In contrast to the related studies described above where data was collected using mimic interfaces

or external sensors, in this study, touch data was not collected using a mimic interface or an external sensor, but was rather collected using an application embedded into the operating system with root privileges.

2.2 Structure of the datasets

The datasets in the related work on authentication have consisted of a variety of touch characteristics. M. Frank et al. (2013) collected a dataset of raw touch events, including the phone's ID, the user's ID, the document being read, time in milliseconds, touch action, phone orientation, x-coordinate, y-coordinate, pressure, size of the area covered and finger orientation. From this set of raw data, 30 features were extracted from each range of raw events into one gesture. In order to simplify the problem down to a binary classification problem, the two classes that were created were the *user of interest* and *all other users*.

Similarly, Li et al. (2013) divided their data into distinct training and testing sets, excluding training users from the testing sets. The raw collected patterns were classified into different gesture types. Each gesture was assigned its own metrics. For example, a tap gesture has only three metrics: the average touch area, the duration of the gesture, and the average pressure. Each type of gesture was then classified using its own classification module. Therefore, to classify all five proposed gestures (sliding right, left, down, up, and tapping), the system needed five classification modules. However, it was shown that the feature extraction process and the classification itself was not detrimental to the phone's performance.

Feng et al. (2012) extracted gesture features from both the touch screen and the sensor glove. Some participants used the sensor glove, while others did not. Collected touch gesture features included coordinates, direction of the motion, speed and distance between multi-touch points. Each gesture was divided into three parts: the beginning, the main motion and the end of the motion.

The study presented in this paper tests authentication using two types of datasets: the unprocessed raw touch data, and processed feature extracted gesture data.

2.3 Classification algorithms used and resulting error rates

The false accept rate (FAR), false reject rate (FRR), equal error rate (EER), and area under the receiver operator characteristic curve (AUC) were used to analyse biometric accuracy in this study.

The FAR is the probability of accepting users who are not who they claim to be. The FRR is the probability of rejecting users who really are who they claim to be. The EER is the point at which the FAR and the FRR are the same, that is, the point at which the threshold for rejecting or accepting a user results in the same number of false acceptances and false rejections. This is shown in Figure 1.

The receiver operator characteristic (ROC) curve is a plot of the true positive rate (TPR) against the false positive rate (FPR) for different threshold values (Bradley, 1997) as shown in Figure 2. The ROC curve is a better indication of an algorithm's accuracy than a single accuracy rate, since varying threshold values can have a large effect on accuracy (Fawcett, 2006). The AUC is used as a single number which represents the ROC curve's shape. It is desirable that the AUC is more than half of the total area.

https://doi.org/10.18489/sacj.v28i2.374

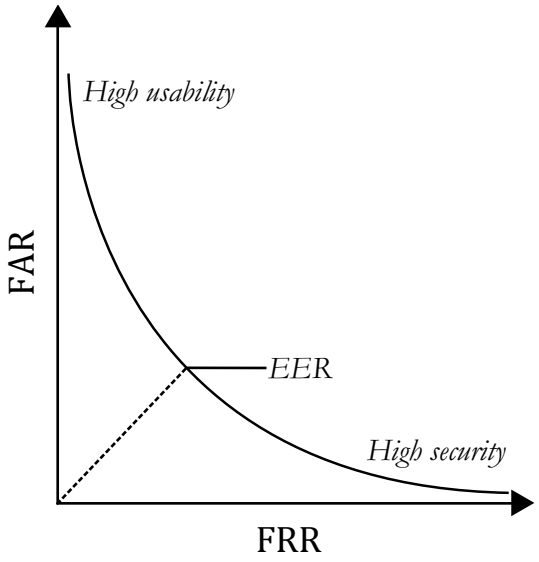

Figure 1: The tradeoff between a high FAR and a high FRR, where the EER is the point at which they are balanced.

Related studies on touch biometrics have used different approaches to reporting on classification accuracy. M. Frank et al. (2013) tested the data using the k-nearest neighbour (k-NN) and support vector machine (SVM) classification algorithms, showing EERs of 0 to 0.4, with the lowest error rates achieved using the SVM algorithm. In the study by Li et al. (2013) the classification using the SVM algorithm also achieved relatively high accuracy rates for all classification modules. Feng et al. (2012) used a set of three classifiers: a decision tree, a random forest, and a Bayes net classifier. Their system achieved a FAR of 0.047 and a FRR of 0.001. These rates are based on the assumption that if three out of seven gestures match the authorised user the user remains authenticated. The authors mention that their system "strives to achieve a low FRR".

Previous work has focused on a variety of algorithms, and most included the SVM algorithm, a decision tree based algorithm or instance-based learning. In this study, the C4.5 (Quinlan, 1993) and K* (Cleary & Trigg, 1995) classification algorithms were used to investigate classification accuracy rates. These accuracy rates are reported in detail and an analysis of EER, ROC curves and the corresponding AUCs are discussed.

The next section discusses the setup of the experiment to investigate whether raw data needs to be processed into gesture data, and whether authentication can be performed using touch biometric data.

3 EXPERIMENT SETUP

Touch data was collected from 30 participants to investigate whether the data is distinct enough to be used for biometric classification. This section describes the experiment protocol, technical details

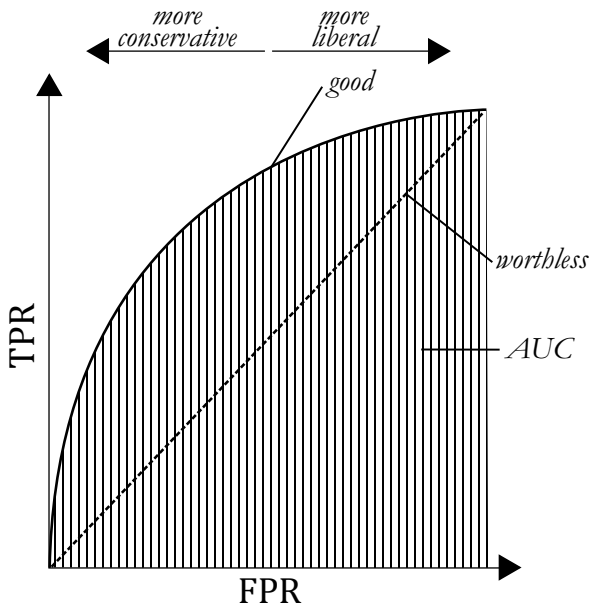

Figure 2: The ROC curve and resulting AUC.

of how the touch data was collected, and how the datasets for authentication were constructed.

3.1 Experiment protocol

The protocol used for collecting touch data was as follows. At the start of each session the nature of the experiment was explained and each participant was asked to read a disclaimer, sign an informed consent form and fill in a brief questionnaire. The questionnaire contained questions about the participants' experience with smartphones and their field of expertise. Most of the participants were from the information technology field and had average to high experience in using smartphones and other devices such as tablets. The participants were then directed to perform six tasks on a smartphone that represented real world interaction with a phone while their touch data was being recorded. Participants were asked to perform six different general tasks on the smartphone representing normal usage of the device. They were not restricted in terms of which touch gestures they could perform or how many should be performed. The usage tasks included reading an article, playing a word matching game and typing a message.

3.2 Data collection

The Android OS was chosen as a platform for the development of a data collection tool because it is open source and customizable. The version of Android running on the phone at the time of the experiment (Android 2.2) was replaced by Cyanogenmod 7, a custom derivative of the Android OS developed by a community of independent developers. It enables users to replace the stock Android

OS running on the phone. Without using Cyanogenmod, the phone's version of Android would have been too old to perform the experiment.

The Android software development kit (SDK) provides a set of motion event features that can be collected to form datasets for touch biometrics. A motion event is triggered whenever a user touches the touchscreen. The motion event contains information such as the pressure applied to the screen and the coordinates of the touch.

The experiment required the development of an application to collect touch event data. The initial approach was to create an application that overlays a transparent screen (or *view* in Android) over all other applications and intercepts touch events. However, Android does not allow applications to do this, since it could be exploited to write keylogger applications. This restriction in gathering touch data from the system has also been highlighted in other research by M. Frank et al. (2013), Li et al. (2013) and Feng et al. (2012), where either the system logs were snooped to collect data, or a mimic interface was created to collect data, as discussed in Section 2.

On Android, only the foreground application that is currently being used is allowed to record detailed touch information. Any application that is overlayed on top of the foreground application is blocked from recording event information. However, some views are exempt from these restrictions because they form part of the system applications. Views with their layouts specified as type *system overlay* may only receive limited touch data (Android Open Source Project, 2013), but views with their layouts specified as *secure system overlay* are allowed to collect any touch information. However, the type *secure system overlay* can only be used by system applications and special permissions are needed. The OS needs to grant permission to the application to use the *secure system overlay* type of layout. Therefore, the OS needs to be altered to allow applications to collect information about touch events.

Android installations usually contain a set of developer tools. Among these tools is an application called "Pointer Location". When activated, this application displays an overlay view on top of other applications and displays the touch data features that it can measure. Pointer Location's window layout is of type *secure system overlay*. Pointer Location outputs data to a private log file, but its output cannot be read by other applications. To collect touch data, the stock Pointer Location application was replaced with an altered version for the experiment. The code to record touch data was hidden within this application. This altered version recorded more features and revealed its output to the system log. In a real-world implementation, the data would rather be stored to the phone's internal database. The application was also made completely invisible to the users, so that it did not display any indication that it was running. The altered OS was compiled and deployed to the phone. Once the Pointer Location application was enabled on the new OS, it could be used to collect any required touch data. This allowed the data collected in the experiment to represent real-world touch interactions, since there was no need for a "mimic" interface or a set of limited actions as has been used by other studies.

https://doi.org/10.18489/sacj.v28i2.374

3.3 Raw and gesture data

Two datasets were created – the first dataset contained the raw data from the Android motion events. The second dataset contained features extracted from the raw data to form gesture data. A gesture is made up of several raw motion events occurring between a down and and up action. This is shown in Figure 3. The features contained in each dataset are listed in Tables 1 and 2.

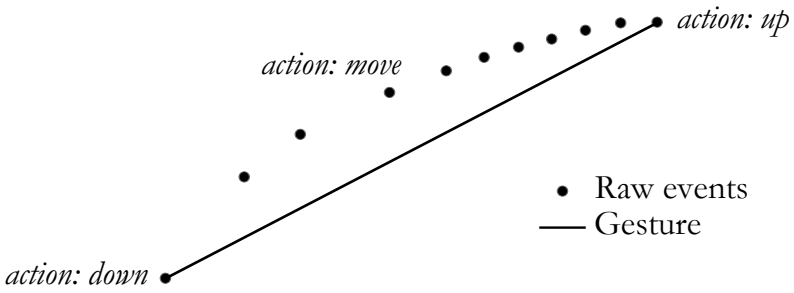

Figure 3: Raw events between a down and an up action form a gesture.

Gesture features listed in Table 2 were calculated using the vector of the movement from the start coordinates to the end coordinates. Other gesture features are the means and standard deviations of raw features. Where the measurement of the feature is a nominal value, the value at the end of the gesture is used. Some features were derived from the raw data for the gesture data. For instance, the inter-stroke time was derived from the times recorded between the last gesture's up action and the current gesture's down action. The vector features refer to the vector between start and end coordinates of the gesture. The last five features in Table 2 were calculated using the vector of the movement from the start to the end coordinates.

The raw feature vector is defined in Equation 1 and the gesture feature vector is defined in Equation 2.

$$(g, t, a, i, x, y, p, s, f_a, f_b, f_c, f_d, f_\theta, s_\theta) \tag{1}$$

$$\begin{aligned}(&g_{end}, i_{gesture}, t_t, t_i, x_s, y_s, x_e, y_e, \overline{x}, x_\sigma, \overline{y}, y_\sigma, \overline{p}, p_\sigma,\\ &\overline{s}, s_\sigma, \overline{f_a}, f_{a_\sigma}, \overline{f_b}, f_{b_\sigma}, \overline{f_c}, f_{c_\sigma}, \overline{f_d}, f_{d_\sigma}, \overline{f_\theta}, f_{\theta_\sigma}, s_{\theta_{end}},\\ &m_\theta, m_d, |m|, \overline{m_v}, \overline{m_a})\end{aligned} \tag{2}$$

The raw dataset contained 14 unprocessed features, while the gesture dataset contained 32 features, excluding the user ID. Although the gesture dataset contained more features than the raw data, the number of instances in the raw dataset was larger than the gesture dataset by a factor of seven. Therefore, on average, one gesture is made up of around seven raw events. The larger size of the raw dataset resulted in more processing time during classification than for the gesture dataset.

https://doi.org/10.18489/sacj.v28i2.374

Table 1: Raw features

Symbol	Description	Units
g	*Type of gesture*: The gesture that the user was busy executing at the point that this *motion event* was triggered.	Tap, double tap, scroll, fling or unknown
t	*Total time*: The time since the initial down action until this motion event was triggered. Resets after each up or cancel action.	Milliseconds
a	*Action type*: The action performed on the screen.	Move, up, down or cancel
i	*Pointer ID*: The pointer ID that is performing this raw event.	0 or more
x, y	*Coordinates*: Most recent x and y coordinates of event.	Integer: > 0 and $<$ screen dimensions
p	*Pressure*: Finger pressure.	Continuous: $0-1$
s	*Size*: Approximate touch area.	Continuous: $0-1$
f_a, f_b	*Touch major and minor*: The length of the major and minor axes of the ellipse that describes the touch area.	Continuous: Device dependent
f_c, f_d	*Tool major and minor*: The length of the major and minor axes of an ellipse that describes an estimation of the actual size of the finger or pen.	Continuous: Device dependent
f_θ	*Tool orientation*: The orientation of an ellipse that approximates the actual finger or pen.	Radians clockwise from vertical
s_θ	*Screen orientation*: The orientation of the screen.	Landscape or Portrait

https://doi.org/10.18489/sacj.v28i2.374

Table 2: Gesture features

Symbol	Description	Units
g_{end}	*Type of gesture*: Complete gesture that the user executed when the gesture is ended (could still be unknown if the system could not determine the type of gesture from the available gesture data).	Tap, double tap, scroll, fling or unknown
$i_{gesture}$	*Pointer ID*: Pointer ID that performed this entire gesture.	0 or more
t_t	*Total time*: Time from the first down action to the last up action.	Milliseconds
t_i	*Inter-stroke time*: Time between the beginning of this gesture and the end of the previous gesture.	Milliseconds
x_s,y_s,x_e,y_e	*Start and end coordinates*: coordinates of the start and the end of the gesture.	Integer: >0 and $<$ screen dimensions
$\overline{x}$,x_σ,$\overline{y}$,y_σ	*Overall coordinates*: Mean and standard deviation of coordinates over entire gesture.	Continuous: >0 and $<$ screen dimensions
$\overline{p}$,p_σ	*Pressure*: Mean and standard deviation of finger pressure over entire gesture.	Continuous: $0-1$
$\overline{s}$,s_σ	*Size*: Mean and standard deviation of approximate touch area over entire gesture.	Continuous: $0-1$
$\overline{f_a}$,f_{a_σ},$\overline{f_b}$,f_{b_σ}	*Touch major and minor*: Mean and standard deviation of the length of the major and minor axis of an ellipse approximating the touch area.	Continuous: Device dependent
$\overline{f_c}$,f_{c_σ},$\overline{f_d}$,f_{d_σ}	*Tool major and minor*: Mean and standard deviation of the length of the major and minor axis of an ellipse approximating the actual finger or pen.	Continuous: Device dependent
$\overline{f_\theta}$,f_{θ_σ}	*Tool orientation*: Mean and standard deviation of the orientation of an ellipse approximating the actual finger or pen.	Radians clockwise from vertical
$s_{\theta_{end}}$	*Screen orientation*: Orientation of the screen at the end of the gesture.	Landscape or portrait
m_θ	*Vector angle*: Angle of the deviation from the closest direction (up, down, left or right).	$0°-45°$
m_d	*Vector direction*: General direction of the motion vector.	Up, down, left or right
$\lvert m \rvert$	*Vector length*: Length of the motion vector.	Integer: >0
$\overline{m_v}$	*Vector speed*: Average speed of the motion vector.	Continuous: >0
$\overline{m_a}$	*Vector acceleration*: Average acceleration of the motion vector.	Continuous: >0

https://doi.org/10.18489/sacj.v28i2.374

3.4 Authentication datasets

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

To create an authentication dataset for all users, each user was in turn treated as the known user (labelled as their assigned user ID), and a set of other users as the unknown users (labelled as unknown). Each training set contained 70% of the genuine user's data and was tested using the other 30% of the data. Formally, each training set included a set K_{train} of known user's data, a set of unknown user's data A and the corresponding testing set included a set K_{test} of known user's data and a set B of unknown user's data, where

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

For every user, a collection of 30 training and testing sets was created from both the raw and the gesture data as described above.

3.5 Limitations of the experiment

For the purposes of this study, the touch biometric system was tested on only one smartphone (HTC Desire) to limit the differences in measurements that could arise if more that one smartphone is used. Furthermore, participants only used the phone in a 20 minute session. Therefore, the effect of using the system for longer was not determined.

Two classification algorithms were used to analyse the data and these were kept relatively simple. The focus of the study was not to determine which classification algorithms performed the best or how to optimise them, but rather to determine whether touch data could be classified accurately for authentication. Further work could involve utilising more different classification techniques, such as neural networks.

https://doi.org/10.18489/sacj.v28i2.374

4 RESULTS

Each dataset was classified using two classification algorithms from the Weka data mining software (Hall et al., 2009). The first algorithm was the J48 tree classification algorithm (E. Frank, n.d.), the Java implementation of the C4.5 tree algorithm (Quinlan, 1993). The second algorithm was the K* classification algorithm (Cleary & Trigg, 1995). Weka default values for parameters were used: For the C4.5 tree algorithm, the confidence factor c was set to 0.25 and the minimum number of objects m was 3. For the K* algorithm, a blending parameter value of 20% was used.

The classifiers were evaluated using the FAR, the FRR, the EER and the AUC. The ROC curve and the AUC were used to give a general impression of a system's performance for different levels of the threshold (strictness) (Bradley, 1997).

4.1 Information gain of touch biometric features

Information gain is used in this section as a measure of the relative importance of each feature in relation to the class. The calculation of information gain was performed using the Weka data mining software. The information gain of attribute a on the set of data instances x, $IG(x,a)$, is calculated as follows:

$$IG(x,a) = H(x) - H(x|a) \tag{3}$$

where $H(x)$ is the entropy of the set of instances x. That is, if splitting the dataset using attribute a results in a low entropy value in relation to the entropy of the set before splitting, the information gain will be high. To determine which were the most informative features in the touch data overall, the raw and gesture dataset features were analysed according to their information gain values with respect to the user's classification of "me" versus "not me" classes.

As described in Section 3.4, the dataset for each user consisted of 30 files representing the known user and different combinations of unknown users for both the raw touch and gesture data. For each of these files, the information gain of each attribute with respect to the user's classification was calculated. To combine this information for each user, a voting approach was used where each file for each user "votes" for the feature with the highest information gain. Figures 4 and 5 show these voting counts for the raw and gesture data respectively. Each user (from 1 to 30) is represented as a column. For each touch data feature in the rows, the value indicates the number of files for which that feature had the highest information gain. For example, in Figure 4 the line of zeros for the top three features indicates that these features did not have the highest information gain in any of the files for any of the users. Therefore, in the raw dataset, the features *gesture type*, *action type* and *pointer ID* seemed to be of less importance than some other features in verifying the identity of the user.

In contrast, a vote count of 30 means that every file that was generated for that user, voted for the same feature as the most informative. A high information gain vote count for a feature would mean that most of the trees generated by the C4.5 classification algorithm would first split on that feature, before using other features as secondary discriminators. For example, in the case of all files for user 10, *x-coordinate* was the most informative, but for user 5, *x-coordinate* was only the most

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

Figure 4: Information gain votes for the raw features for each of the 30 users. Red indicates that no votes were received and green indicates that all files voted for this feature. The remaining values are represented by a colour scaled between red and green.

informative for three of the files, but *y-coordinate* was the most informative for 27 of the files. This means that the horizontal touch positions were a primary distinguishing feature for user 10, whereas the vertical touch positions were a primary distinguishing feature for user 5.

The total column in Figures 4 and 5 is an indication of the relative overall importance of each feature for all users. The five highest total values for the raw dataset were obtained for the features *pressure*, *x-coordinate*, *touch minor axis*, *y-coordinate* and *touch major axis*. This indicates that important distinguishing raw features between users included individual finger pressure, location of touch on the screen (*x*- and *y*-coordinates) and the shape of the finger touching the screen (*touch minor* and *touch major* axes). The location of the touch on the screen was distinguishing because some users concentrated on different parts of the screen, for example by scrolling mostly on the left hand side or more to the top of the screen than other users.

In the case of the gesture features (Figure 5), the *pointer ID* was the most distinguishing feature for most of the users. The pointer ID is a value of 0 or more, where a value of more than 0 indicates that the user used a multi-touch gesture. If only a few participants used multi-touch gestures, this would be a primary feature on which to split the data for distinguishing between users. However, if the sample size is larger with more users utilising multi-touch gestures, the pointer ID may not be as distinguishing. Other features that showed importance for distinguishing users included *pressure mean* and *touch minor mean*, which were also important characteristics in the raw dataset.

From the values in both Figures 4 and 5 it is clear that not all users are classified using the same features. For example, while a number of files in the raw dataset voted for *pressure* as the most informative feature, there were also many users for which the pressure was never the most informative feature. In the case of user 9, the screen orientation was the most distinguishing feature since this user turned the phone to landscape mode when using the touch screen.

Although it is interesting to see which features seem to be important in terms of distinguishing

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

Figure 5: Information gain votes for the gesture features for each of the 30 users. Red indicates that no votes were received and green indicates that all files voted for this feature. The remaining values are represented by a colour scaled between red and green.

between users, it does not answer the main question of whether authentication can be achieved through analysis of touch data. The following section directly addresses this issue.

4.2 Authentication classification accuracy

The accuracies of the classification using the C4.5 tree algorithm and the K* algorithm for the raw and gesture datasets are shown in Tables 3 and 4 respectively. Based on all the measures, the raw dataset had higher classification accuracy rates than the gesture dataset for both algorithms. This could be because the raw features are more informative, but it could also be due to the larger amount of raw data available than feature data on each user. In the gesture dataset, there are more features for each data instance. However, a lot of information is lost during the feature extraction process, when features are aggregated, so the number of data instances is reduced. The effect of the amount of data on accuracy levels is investigated in Section 4.5.

For the misclassified instances, the system generally accepted more imposters than it rejected

Table 3: Accuracy when authenticating users based on the raw datasets. FAR, FRR and percentage correct are reported for a threshold value of 0.5.

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

Table 4: Accuracy when authenticating users based on the gesture datasets. FAR, FRR and percentage correct are reported for a threshold value of 0.5.

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

genuine users (asseen by the higher FAR than FRR). These values are reported for a threshold value of 0.5. The threshold could be adjusted to be stricter for high security applications. For the raw dataset, although the overall percentage of correctly classified instances was slightly higher for the C4.5 algorithm, the K* algorithm performed better overall as reflected in the higher value for AUC. The same pattern emerges for the EER values. Although the FAR was slightly lower for C4.5, the FRR was higher, and the lower EER for K* indicates that there exists a more optimal compromise between FRR and FAR than for the C4.5 algorithm.

The best area under the ROC curve that was achieved was 0.927 for the K* algorithm on the raw dataset and the corresponding equal error rate was 0.148. These rates are worse than reported rates from related work (M. Frank et al., 2013; Li et al., 2013; Feng et al., 2012), but because this study was conducted on real-world touch interactions, these results can be viewed as more realistic. An equal error rate of 0.148 is too high to be used as a high security application since 15% of imposters could be let into the system. However, at least 85% of attackers would need to breach an extra layer of authentication on a smartphone that would otherwise require no authentication. Therefore, for the intended purpose of the system, the accuracy is acceptable, but should be improved in future work.

4.3 ROC curves

ROC curves give an indication of the accuracy of a system across different threshold values, and the AUC is a single number representing that accuracy.

https://doi.org/10.18489/sacj.v28i2.374

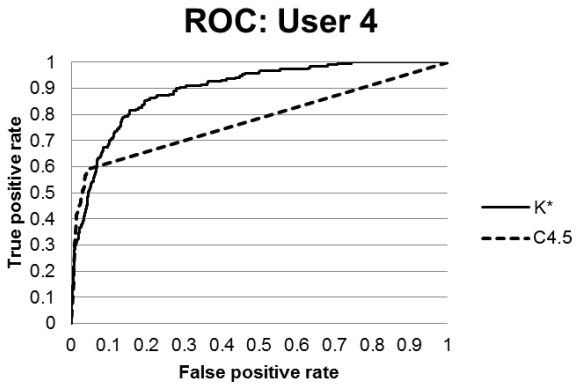

Figure 6: Gesture Dataset

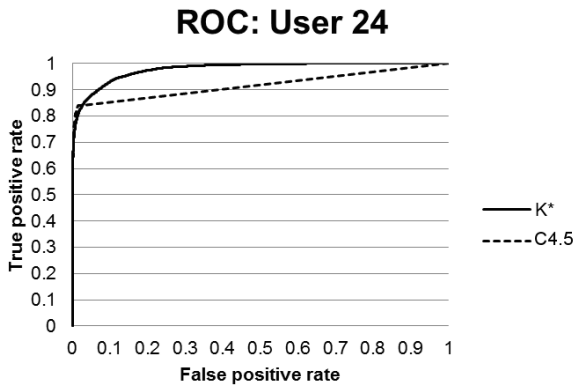

Figure 7: Raw Dataset

According to the AUC value, the K* algorithm generally had better accuracy than the C4.5 tree algorithm. Analysis of the ROC curves shown in Figure 6 for User 4 from the gesture datasets shows that the C4.5 tree algorithm has a less smooth curve between threshold values of 0.5 and 1. (User 4 is a representative of the average AUC for both algorithms in the gesture dataset. The trend was similar for other users.) The C4.5 tree algorithm has fewer points on the curve since some threshold values result in the same predictions, because of the structured nature of the modelled tree. This indicates that it was more difficult to balance the FAR and FRR of the tree than it was for the K* algorithm.

The same pattern is seen to a lesser extent in the raw data set as shown in Figure 7. Here user 24 was chosen as a representative of the average AUC. Therefore, K* algorithm could more easily be used for low or high security systems using any threshold value. However, as shown in the graph, the C4.5 tree algorithm had lower error rates for lower threshold values for both datasets.

For the purposes of the proposed touch biometric system, a more conservative classifier is preferred. However, if users are allowed to adjust the threshold value, then the K* algorithm is preferred since it has a smoother curve and has low error rates for all threshold values. Furthermore, the K* algorithm does not need to be retrained periodically and is therefore more suited to the proposed system. It should be noted, however, that a lazy classifier such as K* takes longer to classify each new instance as the dataset grows, so the dataset would have to be kept to a limited size to be practically applicable for continuous classification.

4.4 Analysis of accuracy across users

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

Figure 8 shows how the classification accuracy as measured by the AUC varies between users. It is clear from this graph that the accuracy depends on the dataset and algorithm used. The same trend is shown inversely for the classification error as measured by EER in Figure 9. Some users could only be classified with good AUC values if the correct algorithm and dataset combination was

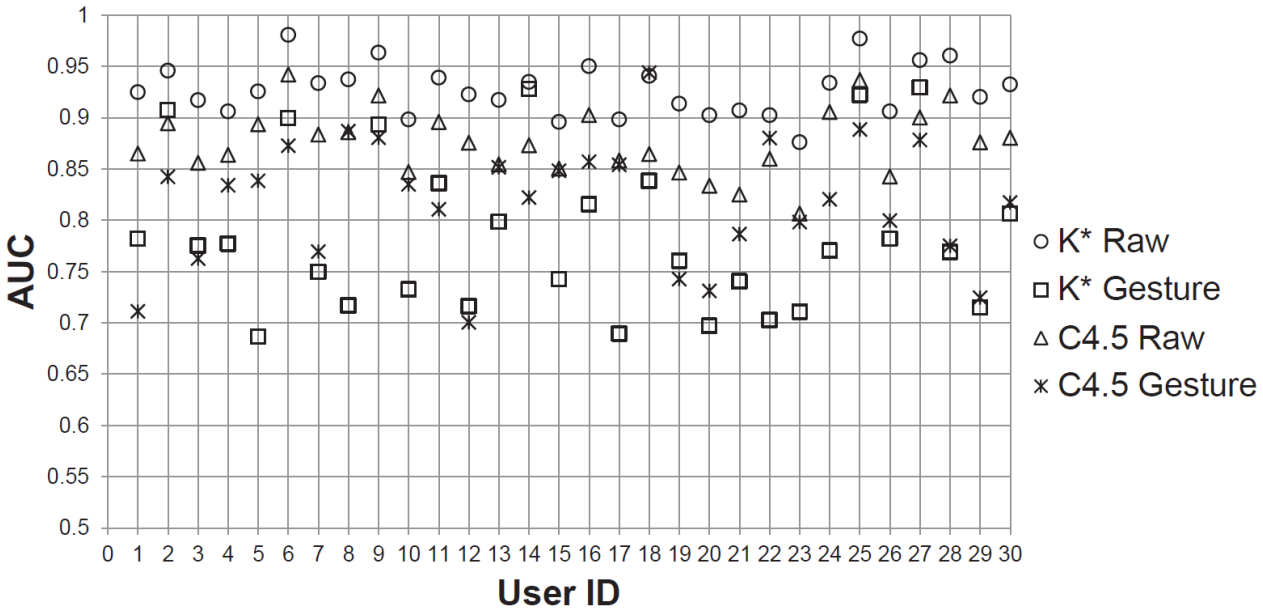

Figure 8: Comparison of the average AUCs for each user ID during authentication.

used. Some algorithms will therefore be more accurate for some users.

It can also be seen in Figures 8 and 9 that some users were generally more recognisable than others, since they had better accuracy rates than other users with any algorithm and dataset combination. For instance, users 6, 9, 18, 25 and 27 were classified with high AUC values in most cases.

Users 6 and 9 turned the phone to landscape when browsing the Internet. This is a distinct behaviour and may have resulted in the high accuracy during classification. The information gain votes of these two users' datasets in Figures 4 and 5 show that screen orientation and tool orientation are important features that arose from this behaviour.

User 14 was observed to have very long finger nails, and complained that the nails interfered with typing on the touchscreen. This resulted in distinct pressure measurements as shown in Figures 4 and 5.

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

User 25 disclosed a touch affecting disability in the experiment questionnaire. The expected system behaviour was that this user would have a very distinct touch pattern. The graph shows that user 25's data results in high accuracy rates across all algorithm and dataset combinations. Figures 4 and 5 show high information gain from the pressure and touch ellipse features.

For some users, the high classification accuracy may only be as a result of generating more touch data by repeating actions or by simply taking longer to complete the tasks. The addition of more

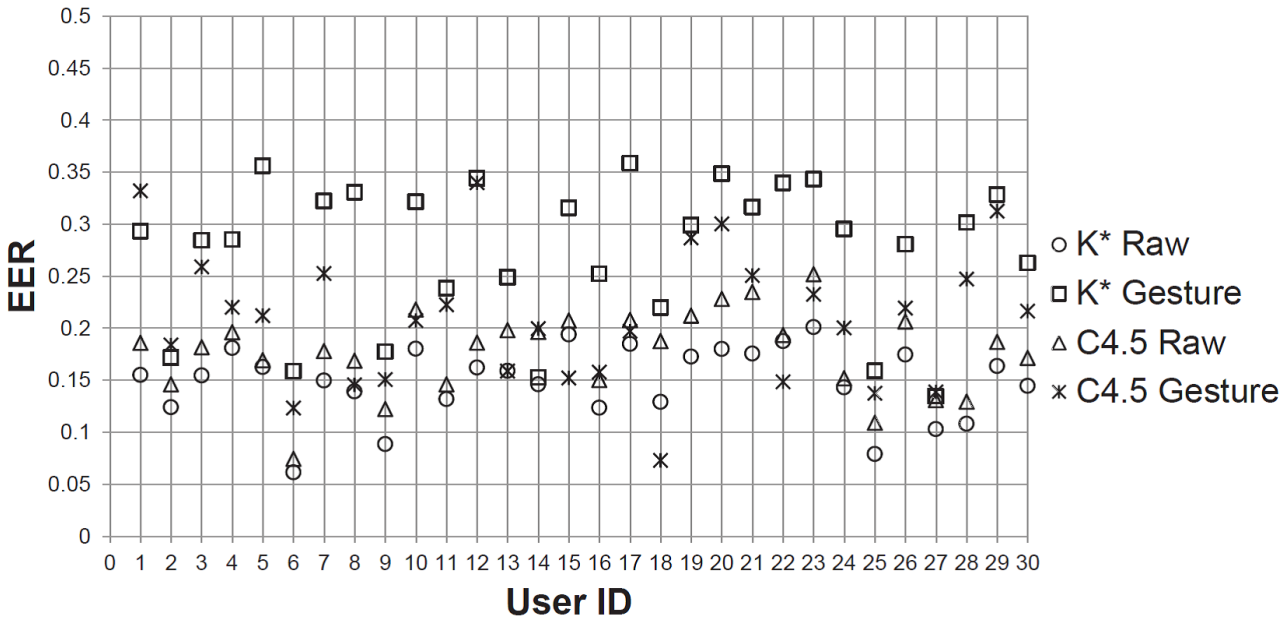

Figure 9: Comparison of the average EERs for each user ID during authentication.

training data may lead to better results. The effect of the amount of training data on the EER is investigated in the next section.

4.5 Effect of the amount of touch data on accuracy and performance

In the experiment the participants were limited to a time period of 20 minutes. Some users completed the six tasks within 10 minutes. Therefore, the amount of data collected from each user was not large in relation to what a real-world system would collect over time from the user of a smartphone. It is suggested that the system will improve over time as the amount of collected data is increased. That is, if the system is used for a long period the FAR and FRR will decrease and the overall security and usability will improve.

Classification was more accurate for some users in the datasets. It is possible that these users performed more or longer gestures, resulting in more data for these users. To investigate this possibility, the datasets were reduced to be no bigger than the size of the smallest dataset: the user that generated the least touch data. Figure 10 shows the result of reducing all datasets to the same size on the raw touch dataset, using the K* classification algorithm. As can be seen, the EER increased for most users.

Figure 11 shows that the same process with the C4.5 tree algorithm resulted in similar increases in EER, except for two user datasets that were classified with fewer errors. This may indicate that overfitting occurred when the classifications were modelled. However, the values are small and do not indicate that this trend would occur reliably in other tests.

Figures 10 and 11 show that decreasing the amount of data available increases errors in the

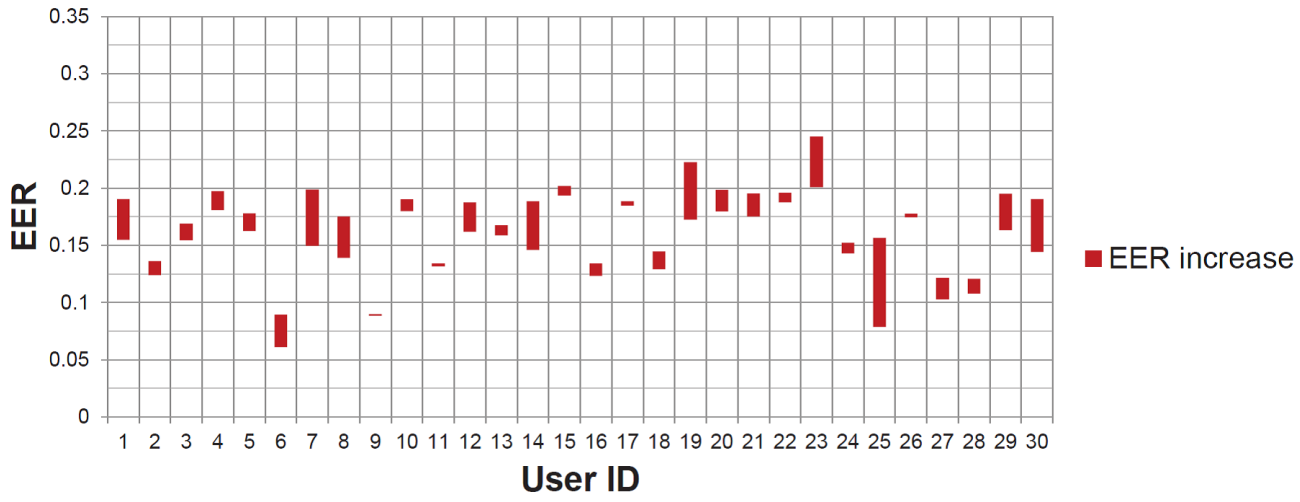

Figure 10: Change in EER for the K* algorithm when all datasets are reduced to the same size.

general case. However, some user datasets always achieve high accuracy, and decreasing the amount of data did not level out error rates across users.

5 CONCLUSION

This study was conducted to investigate whether authentication can be done more easily, securely and continuously on smartphones. It is suggested that one way to implement better authentication on smartphones is through the use of continuous touch biometrics: biometric features measured using touchscreens on smartphones during normal phone use. It was shown that these features could be measured continuously, enabling authentication at all times and creating a constant barrier between the smartphone and attackers.

The biometric value of touch data was tested by collecting touch data using an Android smartphone and then analysing the data in terms of its classification accuracy. This analysis showed that good accuracy rates could be achieved on the touch data. The best accuracy was achieved using the K* classification algorithm, which was an AUC of 0.927 and a correspoding EER of 0.148. Therefore, when the system is balanced in terms of the level of strictness of the threshold value, 15% of attackers will be allowed access, and 15% of legitimate users will be blocked. A balanced application of this algorithm would not be secure or usable enough for certain applications. However, the threshold value could be used to customise the level of security against usability required for a specific application. Limited analysis of different threshold values were performed in this experiment. The threshold value could be changed to improve either the FAR or FRR, depending on the intended application of the system. In an implemented system, a failover approach could be used which prompts the user for a password if the system locks the phone. This will ensure that the security of a phone will not depend solely on the touch biometric authentication mechanism.

https://doi.org/10.18489/sacj.v28i2.374

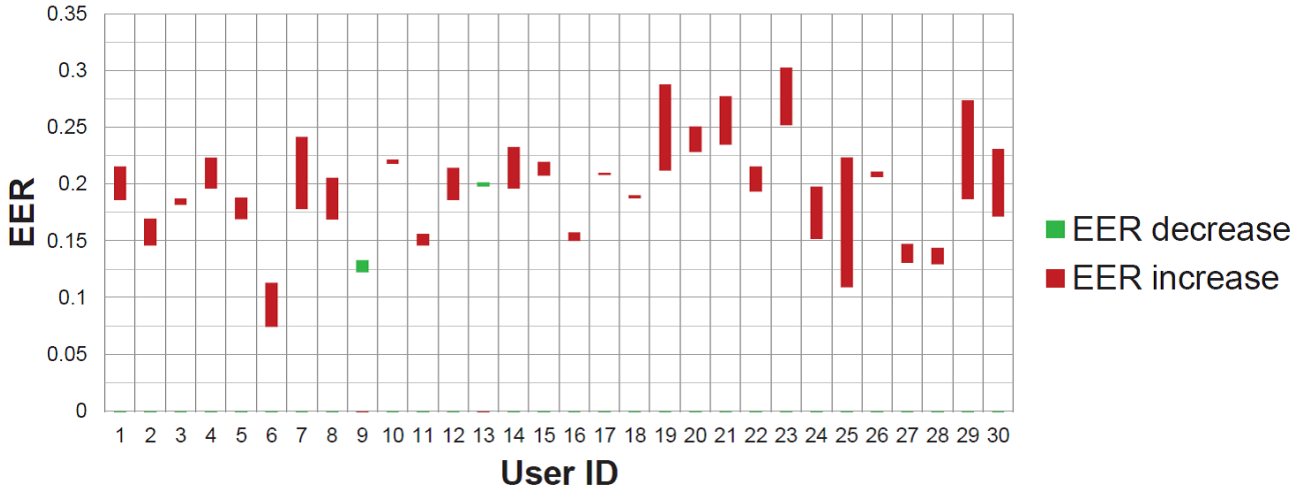

Figure 11: Change in EER for C4.5 algorithm when all datasets are reduced to the same size.

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

High accuracy rates were not caused by some users generating more data. The effect of the amount of data per user is unique to each user. When equal amounts of data per user was tested, the error rates were not equal across different users. Reducing the amount of data resulted in higher error rates, except in two cases, where error rates decreased slightly. This indicates that overfitting could occur early on, and checks should be performed to ensure that the classification algorithm is not overtrained.

This study has shown that continuous touch biometrics can be successfully implemented on smartphones and that such a system would be both secure and usable: secure in the sense that the additional authentication of touch biometrics is continually running in the background and does not require the user to act in a secure way; and usable in the sense that there is no additional cognitive load placed on users to remember particular PINs or patterns.

6 FUTURE WORK

Touch biometrics is a new field and therefore requires further investigation into the effect of various factors on accuracy. These factors could include changing user behaviour due to new environments, new activities or particular social situations. For example, a user may only occasionally use their phone to show photos. The swiping action may not yet have been presented to the authentication

system previously and it may cause the phone to lock.

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

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

References

Adams, A. & Sasse, M. A. (1999, December). Users are not the enemy. *Communications of the ACM*, *42*(12), 40–46. https://doi.org/10.1145/322796.322806

Agrawal, A. (2013). *User authentication mechanisms on Android* (Doctoral dissertation, Indian Institute of Technology, Bombay).

Android Open Source Project. (2013). Windowmanager.layoutparams documentation. Last accessed 23 Nov 2016. Retrieved from http://developer.android.com/reference/android/view/WindowManager.LayoutParams.html

Angulo, J. (2012). *Usable privacy for digital transactions: Exploring the usability aspects of three privacy enhancing mechanisms* (Licentiate thesis, Karlstad University).

Aviv, A. J., Sapp, B., Blaze, M., & Smith, J. M. (2012, December). Practicality of accelerometer side channels on smartphones. In *Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC'12)* (pp. 41–50). Orlando, Florida: ACM. https://doi.org/10.1145/2420950.2420957

Balfanz, D., Durfee, G., Smetters, D. K., & Grinter, R. E. (2004, September). In search of usable security: Five lessons from the field. *Security & Privacy, IEEE*, *2*(5), 19–24. https://doi.org/10.1109/MSP.2004.71

Bonneau, J. (2010). The password thicket: Technical and market failures in human authentication on the web. In *The Ninth Workshop on the Economics of Information Security*. Cambridge, Massachusetts, USA.

Bonneau, J., Herley, C., van Oorschot, P. C., & Stajano, F. (2012, May). The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE Symposium on Security and Privacy (SP)* (pp. 553–567). San Francisco, USA: IEEE.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145–1159. https://doi.org/10.1109/SP.2012.44

Camp, L. J. (2007). Guest editors' introduction: Security and usability. *IEEE Technology and Society Magazine*, *26*(1), 3, 24.

Cleary, J. G. & Trigg, L. E. (1995). K*: An instance-based learner using an entropic distance measure. In *Proceedings of the 12th International Conference on Machine Learning (ICMLA)* (Vol. 5, pp. 108–114). Morgan Kaufmann.

Cranor, L. F. & Garfinkel, S. (2004). Guest editors' introduction: Secure or usable? *Security & Privacy, IEEE*, *2*(7), 16–18. https://doi.org/10.1109/MSP.2004.69

Damousis, I., Tzovaras, D., & Bekiaris, E. (2008, January). Unobtrusive multimodal biometric authentication: The HUMABIO project concept. *EURASIP Journal on Advances in Signal Processing*, *2008*(1), 110:1–110:11. https://doi.org/10.1155/2008/265767

De Luca, A., Hang, A., Brudy, F., Lindner, C., & Hussmann, H. (2012). Touch me once and I know it's you!: Implicit authentication based on touch screen patterns. In *Proceedings of the ACM SIGCHI Annual Conference on Human Factors in Computing Systems (CHI)* (pp. 987–996). Austin, USA: ACM. https://doi.org/10.1145/2207676.2208544

Derawi, M. O. & Bours, P. (2013). Gait and activity recognition using commercial phones. *Computers & Security*, *39*(B), 137–144. https://doi.org/10.1016/j.cose.2013.07.004

Derawi, M. O., Nickel, C., Bours, P., & Busch, C. (2010). Unobtrusive user-authentication on mobile phones using biometric gait recognition. In *Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)* (pp. 306–311). Darmstadt, Germany. https://doi.org/10.1109/IIHMSP.2010.83

Dhamija, R. & Dusseault, L. (2008). The seven flaws of identity management: Usability and security challenges. *Security & Privacy, IEEE*, *6*(2), 24–29. https://doi.org/10.1109/MSP.2008.49

Fawcett, T. (2006, June). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010

Feng, T., Liu, Z., Kwon, K.-A., Shi, W., Carbunar, B., Jiang, Y., & Nguyen, N. (2012). Continuous mobile authentication using touchscreen gestures. In *Homeland Security (HST), 2012 IEEE Conference on Technologies for* (pp. 451–456). https://doi.org/10.1109/THS.2012.6459891

Florencio, D. & Herley, C. (2007). A large-scale study of web password habits. In *Proceedings of the 16th International Conference on World Wide Web* (pp. 657–666). WWW '07. Banff, Alberta, Cananda: ACM. https://doi.org/10.1145/1242572.1242661

Frank, E. (n.d.). J48. Last accessed 23 Nov 2016. WEKA. Retrieved from http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html

Frank, M., Biedert, R., Ma, E., Martinovic, I., & Song, D. (2013). Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *Information Forensics and Security, IEEE Transactions on*, *8*(1), 136–148. https://doi.org/10.1109/TIFS.2012.2225048

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, *11*(1), 10–18. https://doi.org/10.1145/1656274.1656278

Ho, C. C., Eswaran, C., Ng, K.-W., & Leow, J.-Y. (2012, December). An unobtrusive Android person verification using accelerometer based gait. In *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia (MoMM '12)* (pp. 271–274). Bali, Indonesia: ACM. https://doi.org/10.1145/2428955.2429007

Kuseler, T., Lami, I. A., & Al-Assam, H. (2013). Location-assured, multifactor authentication on smartphones via LTE communication. In *Mobile Multimedia/Image Processing, Security, and*

Applications (SPIE) (Vol. 8755, 87550B). Baltimore, Maryland, USA. https://doi.org/10.1117/12.2018027

Li, L., Zhao, X., & Xue, G. (2013). Unobservable re-authentication for smartphones. In *Network & Distributed System Security Symposium (NDSS)*. San Diego, USA.

NTT. (2003). Docomo's newest 505i handset features fingerprint authentication. Last accessed 23 Nov 2016. Retrieved June 14, 2012, from https://www.nttdocomo.co.jp/english/info/media_center/pr/2003/000985.html

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann.

Sae-Bae, N., Ahmed, K., Isbister, K., & Memon, N. (2012). Biometric-rich gestures: A novel approach to authentication on multi-touch devices. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems* (pp. 977–986). New York, NY, USA: ACM. https://doi.org/10.1145/2207676.2208543

Saevanee, H. & Bhatarakosol, P. (2008). Authenticating user using keystroke dynamics and finger pressure. In *2008 International Conference on Computer and Electrical Engineering (ICCEE)* (pp. 82–86). International Conference on Computer and Electrical Engineering ICCEE. Phuket, Thailand: IEEE. https://doi.org/10.1109/ICCEE.2008.157

Sasse, M. A., Brostoff, S., & Weirich, D. (2001). Transforming the 'weakest link'–A human/computer interaction approach to usable and effective security. *BT Technology Journal*, *19*(3), 122–131. https://doi.org/10.1023/A:1011902718709

Shahzad, M., Liu, A. X., & Samuel, A. (2013). Secure unlocking of mobile touch screen devices by simple gestures: You can see it but you can not do it. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking (MobiCom)* (pp. 39–50). MobiCom '13. Miami, Florida, USA: ACM. https://doi.org/10.1145/2500423.2500434

Silverman, D. (2012). Android 4.0's facial recognition is cool, but don't trust it yet. Retrieved from http://blog.chron.com/techblog/2011/12/android-4-0s-facial-recognition-is-cool-but-dont-trust-it-yet/

Smetters, D. K. & Grinter, R. E. (2002). Moving from the Design of Usable Security Technologies to the Design of Useful Secure Applications. In *Proceedings of the 2002 Workshop on New Security Paradigms (NSPW)* (pp. 82–89). NSPW '02. Banff, Alberta, Canada: ACM. 10.1145/844102.844117

Vildjiounaite, E., Mäkelä, S.-M., Lindholm, M., Kyllönen, V., & Ailisto, H. (2007). Increasing security of mobile devices by decreasing user effort in verification. In *Second International Conference on Systems and Networks Communications (ICSNC)* (pp. 80–86). Cap Esterel, French Riviera, France. https://doi.org/10.1109/ICSNC.2007.44

Whitten, A. & Tygar, J. (1999). Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *8th USENIX Computer Security Composium* (pp. 169–184). Washington, D.C., USA: USENIX Association.

Woodward, J. D., Orlans, N. M., & Higgins, P. T. (2003). *Biometrics* (1st ed). Rsa Press Series. Berkeley: McGraw-Hill/Osborne.

Yan, J., Blackwell, A., Anderson, R., & Grant, A. (2004). Password memorability and security: Empirical results. *Security & Privacy, IEEE*, *2*(5), 25–31. https://doi.org/10.1109/MSP.2004.81

Yee, K. (2004). Aligning security and usability. *Security & Privacy, IEEE*, *2*(5), 48–55. https://doi.org/10.1109/MSP.2004.64

Zheng, N., Bai, K., Huang, H., & Wang, H. (2012). *You are how you touch: User verification on smartphones via tapping behaviors*. College of William & Mary Department of Computer Science.