The Journal of Systems and Software 149 (2019) 158–173

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Touch gesture-based authentication on mobile devices: The effects of user posture, device size, configuration, and inter-session variability

Zahid Syed[a,*], Jordan Helmick[b], Sean Banerjee[c], Bojan Cukic[d]

[a] University of Michigan- Flint, USA
[b] West Virginia University, USA
[c] Clarkson University, USA
[d] University of North Carolina at Charlotte, USA

ARTICLE INFO

Article history:
Received 16 November 2017
Revised 8 July 2018
Accepted 13 November 2018
Available online 20 November 2018

Keywords:
Mobile security
Behavioral biometrics
Soft biometrics
Touch based authentication
Continual authentication
Gesture based authentication

ABSTRACT

Touch dynamics is a behavioral biometric that authenticates users by analyzing the characteristics of the touch gestures executed on mobile devices. Current research in this field has mostly focused on identifying the best algorithms and attributes to improve authentication performance. However, such systems must also be resilient against environmental variables.

In this paper, we demonstrate that the user's posture, device size and configuration have a significant impact on the performance of touch-based authentication systems. Our results indicate that authentication accuracy increases with the device size. Furthermore, we conclude that using a device's 3-D orientation is necessary to attain better authentication performance. Our findings indicate that the features used in state-of-the-art touch-based authentication systems are insufficient to provide constant, reliable performance when any of the studied environmental variables change. With this paper, we release a new data set. Unlike the currently publicly available touch-based authentication datasets, our collection protocols control for all the studied variables. Our research study demonstrates threats to validity that noisy environmental conditions introduce to these currently available public datasets.

This work is an extension of a previous publication. Presented user authentication approaches are unique and may have immediate benefits to the development of better touch-based authentication systems.

1. Motivation

Portable touchscreen devices are now ubiquitous. As of 2017, 95% of the U.S. population owns a smartphone and more people rely on smartphones for online access than on desktop computers (Pew Research Center, 2017). By the year 2018, it is projected that more than 50% of users will use smartphones for primary online activity (Gartner, 2013).

However, with the growth in portable device usage, there has been a parallel rise in crimes related to cell phones. Nationwide, 30–40% of all robberies are related to cell phone and tablet theft (Federal Communications Commission, 2012). The motives for such crimes are two-fold: the value of the physical devices themselves, and the personal information stored on these devices. Despite the increasing value of information on such devices, static one-time authentication measures such as PINs or more recently fingerprints, are still universally employed on smart phones and tablets for data security.

Touch screen based mobile devices use touch presses and touch gestures as input for interaction with the user interface. Since touch serves as the primary mode of input on these devices, a natural means to authenticate the user's identity is via touch dynamics, by analyzing the touch gesture behavior of a user. Unlike PINs and passwords which are requested at the beginning of the session, touch dynamics can be used for continual authentication, i.e., the user is actively authenticated throughout the usage session based on his/her pre-enrolled touch dynamics profile.

However, for real-world deployment, such a system must take multiple factors into account to develop an accurate touch dynamics based user profile. These factors include the user's posture, screen size, screen orientation, device specifications, type of app being used, and gesture direction. Creating a user profile based on these factors may lead to a more accurate user model and, thus,

* Corresponding author.
E-mail addresses: zahsyed@umflint.edu (Z. Syed), bcukic@uncc.edu (B. Cukic).

https://doi.org/10.1016/j.jss.2018.11.017
0164-1212/

better authentication performance. At the same time, it would require less data for user model generation.

This work investigates the effect of three factors on the authentication performance of a touch-based authentication system: user posture, device size and device configuration. By user posture, we refer to a combination of the device orientation (portrait or landscape) and the position of the device w.r.t the user (held in hand, placed on table). Each of these factors has considerable potential to affect the touch dynamics profile of a user and thus, the authentication performance. This work is an extension of a previous work (Syed et al., 2015). We conducted an experimental data collection designed to control each factor (posture, device size and device configuration). Using the collected dataset, we determined their effect on the authentication performance of our touch-based authentication system. The results of these tests will allow us to determine those factors that must be accounted for when developing a user model. For example, if changing the user posture is found to have a significant impact on the classifier performance, it would be advisable to develop robust models that can account for different postures.

This paper is structured as follows: Section 2 discusses the current state of the art in touch dynamics research. Section 3 describes the design of the dataset and data collection protocols. Section 4 describes the pre-processing carried out such as noise removal and dimensionality reduction. Section 5 describes the procedure to sample train and test sets for the purposes of the experiments. Sections 6, 7 and 8 answer various questions pertaining to touch dynamics based on an empirical analysis of the dataset. In Section 9, we discuss the technical hurdles when transforming features when porting user models across devices of different sizes and configurations. We also discuss the threats to validity that current public datasets in this domain may contain due to these discoveries. Section 10 discusses the tradeoffs our system offers between authentication accuracy and time to authentication. Section 11 provides a critical discussion of the assumptions of the dataset and the threats to the validity of the conclusions. Finally, Section 12 summarizes the paper and Section 13 concludes with a discussion of future directions for research.

2. Related work

As the field has matured, researchers have delved into various aspects of the touch based authentication problem. These include the different types of gestures such as swipes (Mondal and Bours, 2015b; 2015a), flicks (Nohara and Uda, 2016; Lin et al., 2012), pinches, and slides (Lu and Liu, 2015). A few efforts have extended touch gestures to include motion-based features (Shih et al., 2015).

For feature extraction, researchers use mobile device-specific operating system calls to log the x and y-coordinates, pressure, and timestamp of the atomic components of each gesture. These are then used to derive features such as gesture speed, direction, finger area, correlation values in multi-finger gestures, average duration of the gesture, and other relevant metrics that describe the gesture (Saevanee and Bhatarakosol, 2008; Xu et al., 2014a; Cai and Chen, 2011; Nixon et al., 2016; Nader et al., 2015).

One of the earlier studies into touch-based authentication using modern mobile devices was by Frank et al. (2013) who authenticated users by analyzing their regular device use patterns over time, through 34 different features extracted from touch strokes. This study achieved an equal error rate between 0%-4%. The authors also tested for inter-session authentication that showed that touch-based authentication can be used for long term authentication.

Li et al. (2013) evaluated the performance of a live implementation of a smart phone-based touch authentication system. The touch data was collected in the background from 75 users who were asked to freely use the devices for a number of days. The collected data was used to create a SVM-based classifier that exhibited an equal error rate of 3%.

Miguel-Hurtado et al. proposed gender prediction by using swipes. Their approach utilized various gesture statistics such as length, width, height, pressure, speed, acceleration, arc distance, and start-to-end angle (Miguel-Hurtado et al., 2016). Male and female users were found to exhibit variation in features such as width, area, angle, and speed. Additionally, the authors found that the multilinear logistic regression classifier was most reliable for gender prediction, obtaining 71% accuracy based on an individual swipe direction (down-to-up).

Palaskar et al. (2016) analyzed user habituation and its effects using the same dataset as this work while controlling for device size and user posture. They showed that constantly retraining the classifier is vital to maintaining the classifier accuracy as the user profile continuously changes even after 4 weeks of usage.

In a manner similar to keystroke dynamics, touch gestures can incorporate accelerometer measurements due to device movement when interacting with the screen (Nohara and Uda, 2016; Jain and Kanhangad, 2015).

With reference to reporting benchmark accuracy in touch-based authentication, researchers have used diverse experimental setups that vary significantly with respect to number of users, samples per user, and device(s) used. The range of controlled and confounding factors hampers empirical comparisons. Frank et al. achieved an EER of 2–3% immediately after the experimental session which dropped to 4% after a week. Serwadda et al. (2013) provided a benchmark analysis for touch based user authentication and showed that a Logistic Regression based classifier offers the best results (10.5–17.8% EER depending on device position and user stroke direction). Syed et al. (2015) investigated the effect of device size and user posture and concluded that both significantly affect the classifier performance. They reported mean EERs of 3.8%–8.81% depending on the scenario under consideration. Shen et al. (2015) achieved FAR and FRR of 7.52% and 5.47%, respectively, when the user performed unconstrained tasks, while in application specific tasks the FAR and FRR reduced to 4.68% and 1.17%. They also determined that user's touch behavior exhibits less variability in application specific tasks as compared to unconstrained tasks.

Additionally, Xu, et al. achieved an EER of 10% using SVMs on a dataset collected from 32 users, where 29 of the users provided 200 strokes while 3 provided 1200 strokes (Xu et al., 2014b). Zhang et al. used SVMs and sparity based classifiers on a dataset of 50 subjects, and obtained EERs of less than 15% while using 27 features (Zhang et al., 2015). Shen et al. used SVMs, kNNs, random forest, and neural networks on a dataset of 71 subjects. Each subject provided 2002 strokes, and an EER of 1% is reported for task specific activities and 5% for freeform activities (Shen et al., 2016a). Work by Mahbub et al. collected 3482 strokes from 48 subjects to achieve an EER between 22.1% and 38% when using kNNs, SVMs, Gradient Boosting Models, and random forest (Mahbub et al., 2016). Sitova et al. reported an EER of less than 16% on a dataset comprising of 90 users when using SVM and distance based classifiers, such as scaled Manhattan and scaled Euclidean (Sitová et al., 2016).

Research also suggests that due to the behavioral and unconstrained nature of touch gestures, data is further affected by the dominant hand, mobility, usage changes over time, and user location (Feng et al., 2014). The research literature further suggests that touch data is application-dependent, such that researchers should not expect generalized performance across applications, but should instead consider "context-aware" implementations (Feng et al., 2014; Shen et al., 2016b; Khan et al., 2014).

Table 1
Characteristics of the four devices used in this study. All devices were based on the Android v4.1 operating system.

ID	Manuf.	Model	Type	Display Size	Resolution	Pixel density
T10	Samsung	Tab 210″	Tablet	10″	1280 × 800	149 ppi
T7	Samsung	Tab 27″	Tablet	7″	1024 × 600	170 ppi
S3	Samsung	S3	Smartphone	4.8″	1280 × 720	306 ppi
EVO	HTC	Evo 4G LTE	Smartphone	4.7″	1280 × 720	312 ppi

In addition to our dataset, there are four other publicly available datasets. These are compared to ours in Section 3.4.

3. dataset design[2]

3.1. Device selection

Four devices were selected for the purposes of this study: two tablets and two smartphones with varying screen sizes. Their characteristics are described in Table 1. One smartphone was from a different manufacturer than the remaining devices (2 tablets and another smartphone). Furthermore, the two smartphones, though from different manufacturers, were similar in size and hardware/software configuration. These devices were chosen specifically to answer our research questions. Not all devices are used in every part of the analysis; the details will be discussed in the appropriate sections. All devices ran the same version of the Android operating system. We chose Android for multiple reasons: As of early 2018, Android runs on 85% of worldwide smartphones (IDC, 2017). Thus, any results we obtain on these devices would be applicable to a large segment of the mobile devices. Furthermore, Android's developer, Google, provides a customized Eclipse IDE. This allows for rapid development, prototyping and testing of apps (Android SDK, 2013).

3.2. Design of the data collection app

We created a custom app to capture the natural horizontal and vertical strokes from the user. The app consisted of a photo matching game where the objective was to search through a list of images to find a randomly selected image that the app displayed as an inset. This activity required the user to scroll vertically and swipe horizontally through a randomized image list to find the desired image.

The activities within one game are shown in Fig. 1. When the user begins the game, on Screen 1, he/she attempts to find the displayed photo by scrolling vertically through a randomized list of images. Once the matching photo is found, the user clicks it. Screen 2 is then displayed with another image that must be found. In this step, however, the user must swipe sideways to find the displayed photo. Once the matching photo is found, the user clicks the 'Back' button which causes Screen 1 to reappear with another randomly selected image. This pair of activities repeat five times. The game ends after the fifth iteration.

As the user interacts with the app, the touch data is collected in the background. The structure and type of touch data collected is explained later in Section 4.

3.3. Data collection procedure

As part of the data collection, we recruited 31 test subjects. Each subject took part in at least 8 sessions spaced over 2–3 weeks. In each session, the subject used the app on each of the four devices (refer Table 1) in the following postures:

1. Posture #1 (P1): Device lying flat on a table in portrait orientation
2. Posture #2 (P2): Device held in portrait orientation
3. Posture #3 (P3): Device held in landscape orientation

Since there are 4 devices and 3 postures, each subject played 12 games in every session . A 'game' refers to the subject's interaction with the app on a specific device in a specific posture in one session. Thus, in each session, the subject played a game on one of the devices (in postures P1, P2, P3), then on the remaining three devices (in all postures) in no particular order. Each game typically required 3 minutes to complete.

We chose to conduct 8 sessions per subject based on the results of a pilot study with 7 subjects. Through that study, we concluded that training a classifier on more than 4 games worth of data did not significantly improve the classifier performance. Thus, to collect adequate data for training a classifier and testing it, we conducted 8 sessions per subject. The details of this pilot study are provided in Section 5.4.

3.4. Comparison of our dataset to other publicly available datasets

Based on our literature survey, there are four other publicly available datasets in this field. Table 2 compares the features of these datasets to ours. Please refer to Section 9.3 for further analysis of the limitations and threats to validity of touch-based authentication algorithms stemming from collection limitations inherent in each publicly available data set.

The collection protocol used to create our dataset allows for efficient evaluation of hypotheses related to screen size (phones & tablets), manufacturer, user posture (in-hand vs on a flat surface), screen orientation, intra-session variability, and inter-session variability. This design framework which controls the factors and includes replication of participants, improves statistical validity, reliability, and replicability of the inferences. Although other works have contained subsets of these factors in various forms, none have exposed all participants to all of these factors in a controlled setting.

The average number of strokes collected per user (19373 strokes) is approximately 6 times larger than the next largest dataset. This number is the total strokes per user for all devices and postures combined. Some users may have more strokes than others. If the dataset is pruned so that all users have the same number of strokes in every device-posture combination, the total number of strokes per user is 13080. Please refer to Section 4 for more information on data pre-processing.

4. Data pre-processing

4.1. Noise removal

As the user interacted with the app, the touch data was collected via methods provided in the standard Android library. Before performing feature extraction, we removed certain types of strokes from the dataset that could not be used by the touch-based authentication system. The omitted strokes include:

[2] instructions to download the dataset can be found on this webpage.

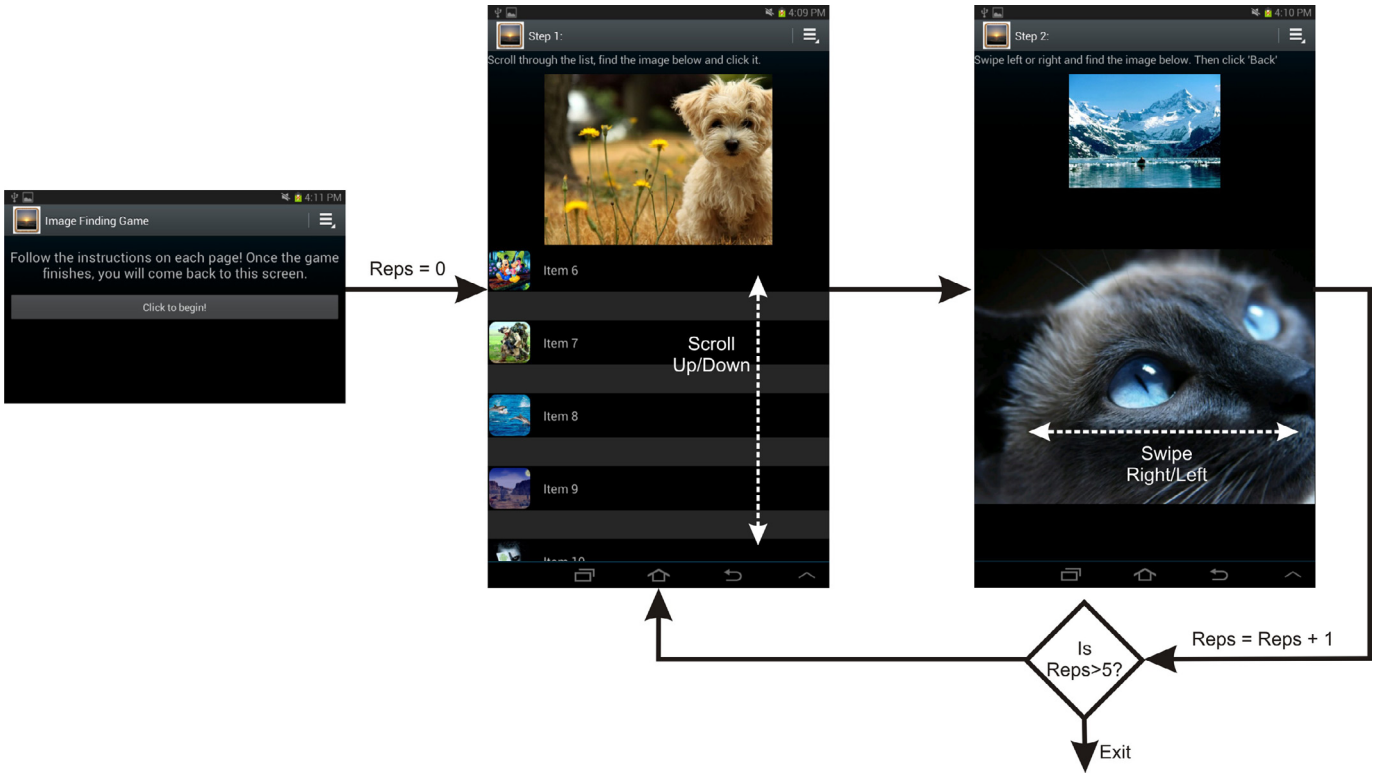

Fig. 1. Activity flow of a single game in the image-matching app.

Table 2
Comparing key characteristics of our dataset to currently available public datasets.

	Characteristics	Frank et al. (2013)	Serwadda et al. (2013)	Antal et al. (2015)	Mahbub et al. (2016)	Syed et al.
Users characteristics	*Users*	41	190	71	48	31
	Gender distribution	N/A	N/A	56M, 15F	36M, 12F	19M, 12F
	Age range	N/A	N/A	19–47	22–31	14–38
	Freq of mobile use	N/A	N/A	4 classes	N/A	4 rarely, 27 daily
	Dominant hand	N/A	N/A	N/A	N/A	11 Left, 20 Right
Dataset characteristics	*Features*	27	28	15	24	18
	Strokes per user	488	400	3260	2482	19373*
	Applications	2	2	2	Free use	1
Experimental controls	*Screen orient.*	2	2	Not Stated	Not Stated	2
	Controlled?	no	No	No	No	Yes
	Devices used by each user.	1 of 5 phones (4 diff manuf)	1 of 1 (mult identical devices used)	1 of 8 (diff. size and res.)	1 of 1 (mult identical devices used)	4 of 4
	Controlled?	No	Yes	No	Yes	Yes
	Posture controlled?	No	No	No	No	Yes (in-hand, on-desk)
	Num of sessions (for inter-session analysis)	2	2	>1	248	≥8

* This is the average total strokes per user for all devices and postures combined in the unpruned dataset.

1. Those that changed direction mid-way.
2. Touch presses (as they have no associated movement component).

Strokes of all lengths were kept in the dataset. While one set of researchers have shown that very short strokes possess less discriminatory power (Frank et al., 2013), we chose to keep them in the dataset and let the classifier decide upon their effectiveness.

The dataset was further pruned so that all subjects had an equal number of horizontal and vertical strokes for any given device and posture combination. This pruning removes any confounding effect caused by some users having a larger number of strokes in the dataset than others. The final dataset consisted of 13,080 strokes per subject with a ratio of about 5:1 between horizontal and vertical strokes.

4.2. Feature extraction and dimensionality reduction

Once the dataset was pre-processed in the manner explained, we extracted 18 features for each gesture. These are listed in Table 3.

Using the test bench setup described later in Section 5, we evaluated the effectiveness of the features used to construct the user model for randomly sampled users. An Information Gain evaluator was used to determine the discriminatory power of a feature and the attributes were ranked according to their individual

Table 3
List of features generated and their description.

Feature	Description
StartX, StartY, StartPressure, StopX, StopY, StopPressure	Abscissa, ordinate and pressure at the location where the gesture began/ended
StrokeDuration*	Duration of stroke (in μs)
Length_EE, Angle_EE*	Distance and angle between beginning and end point (in pixels)
Length_Trj	Length of gesture's trajectory
Ratio_Trj2EE*	This ratio between Length_EE and Length_Trj. This is a measure of deviation of the gesture from a straight line.
AverageVelocity	The average velocity of the gesture
InterStrokeTime	Delay between successive strokes
MidPressVal	Pressure at the midpoint of the gesture
Vel20, Vel50, Vel80	Average velocity after 20/50/80% of the stroke has been executed
Direction*	Primary direction of the stroke (Horizontal/Vertical)

* Discarded in our analysis.

Table 4
Effectiveness of features used to generate user models.

Info. Gain	Feature	Info. Gain	Feature
0.4072	StartX	0.061	StartPressure
0.3654	StopX	0.051	PairedVel20
0.1891	StopY	0.0496	PairedVel80
0.1301	MidPressVal	0.0182	StopPressure
0.109	AverageVelocity	0	Ratio_TrjLen2EELen
0.0923	StartY	0	StrokeDuration
0.092	Length_Trj	0	Angle_EE
0.0906	Length_EE	0	Direction
0.0769	PairedVel50		
0.0706	InterStrokeTime		

evaluations. The mean results of this test are shown in Table 4. The four features that did not contribute any information were pruned from the dataset. Thus, the final dataset contained 14 features.

5. Testbench design

Prior to the experimental analysis, we set up a standardized procedure to sample strokes to create train and test sets, build a user model and generate performance metrics. This enabled us to uniformly sample the dataset and minimize the effect of noise and variations on our experimental results. This procedure is a six step process illustrated in Fig. 2 and described in the remainder of this section:

For any given user, there are two categories of stroke data available: genuine and impostor. The genuine data is the list of all gestures executed by the user throughout the study. As for the impostor data, this is the list of gestures executed by all other users (who act as impostors).

5.1. Sampling strokes to generate genuine and impostor train-test sets

- **Step 1**: To generate the genuine part of the train-test sets, the entire genuine dataset is split into groups of five strokes (illustrated in green in Fig. 2). Let the total number of such groups formed be $2L$. The genuine train and test sets are formed by collating alternating groups of strokes. We sampled groups of strokes for the following reasons:
  - We made the reasonable assumption that touch gestures executed in the adjacent periods of time are similar to each other. This would lead to a more stable user model as compared to sampling individual strokes across the dataset.
  - If a single contiguous block of strokes from one point in time is employed as the train set, it may not adequately represent the user's behavior over the entire data collection period. Sampling evenly throughout the entire dataset eliminates this problem.
- **Step 2**: To create the impostor part of the train and test sets, a total of $2L$ groups are sampled from the impostor dataset, i.e. the same number as in the genuine dataset. For each of the 30 impostors, $2L/30$ groups are sampled evenly across his/her dataset (illustrated in Step 2 in Fig. 2). This helps account for the effects of habituation that may change that impostor's behavior over time.
- **Step 3**: The complete train set is formed by combining the genuine train set and the impostor train set. Similarly, the complete test set is created by combining the genuine test set and impostor test set.

5.2. Generating a user model

- **Step 4**: A Random Forest based classifier models the user's gesture patterns by training from the training set. We chose Random Forest for our analysis based on a benchmark comparison of five classification algorithms on a pilot dataset: Support Vector Machine, Logistic Regression, Naive Bayes, Random Forest and Multilayer Perceptron. Our analysis showed that Random Forest provided the best and most reliable performance. The details of this benchmark study are described in Section 5.5. The particular classification algorithms were selected as they performed best in a benchmark comparison by Serwadda et al. (2013). Every Random Forest classifier in this work has an ensemble of 100 trees and selects the 8 best features during model creation as this combination was found to offer the best performance.

5.3. Calculating performance metrics

- **Step 5**: Each group of five strokes in the test set is evaluated by the classifier and every stroke is assigned a class confidence value by the classifier. This is a measure of the classifier's certainty in predicting that a stroke belongs to a certain class (genuine or impostor). It varies between 0 and 1. For example, a class confidence value of 0.7 for class 'genuine' indicates that the classifier has 70% confidence that the stroke belongs to class 'genuine'.

- **Step 6**: To generate performance metrics, first, a class confidence threshold is set. If a stroke's class confidence value is greater than the threshold, it is assigned the class label 'genuine' (or impostor otherwise). Within a group of strokes, the majority class label is assigned to all gestures in the group. Thus, if the five strokes in a group have 'genuine' class confidence values 0.30, 0.79, 0.65, 0.91, and 1.00, then based on a class confidence threshold of 0.60, all strokes in the group will be assigned the majority class label 'genuine'.
Once all strokes in the test set have been labeled, the False Accept Rate is calculated as,

$$\frac{\#\text{ of impostor strokes classified as genuine}}{\text{Total }\#\text{ of impostor strokes}}$$

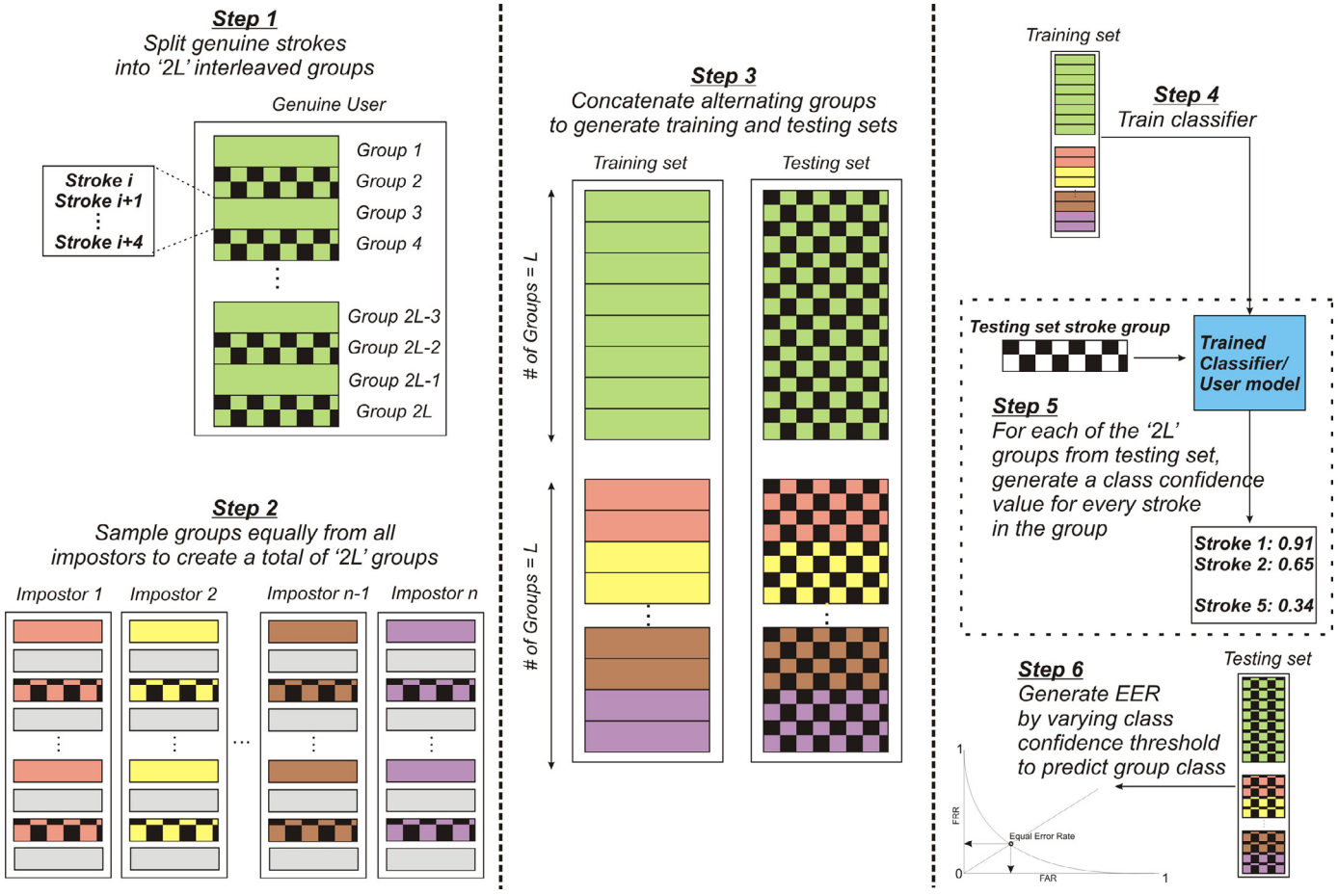

Fig. 2. The procedure for forming train and test sets from the touch data, training the classifier, and generating performance metrics. Each rectangular blocks represents a group of five gestures. The plain colored blocks represent groups used to form the train sets while the patterned blocks represent groups used to form the test sets.

while the False Reject Rate is calculated as,

$$\frac{\#\text{ of genuine strokes classified as impostor}}{\text{Total }\#\text{ of genuine strokes}}$$

The class confidence threshold can be lowered to make the classifier user-friendly and less trustworthy (with low FRR and higher FAR). It can also be increased to make the classifier highly restrictive and more trustworthy (with high FRR and low FAR).

- At a certain class confidence threshold, the FAR is equal to the FRR. This FAR/FRR value is called the Equal Error Rate (EER). The EER serves as a single-valued performance metric for the classifier. In our experiments, the EER was chosen as the default performance metric unless otherwise noted.

5.4. Determing the size of the optimal training set

Before the full fledged data collection, we conducted a pilot study with 7 users on the T7 tablet. This preliminary study was conducted to determine the amount of training data necessary to create an adequate user model. This in turn would help us define the number of sessions we needed to conduct per user in the actual data collection. To answer this question, each of the 7 users took part in 10 sessions. We reiterate that a session refers to playing 3 games on each of the four devices with each of the 3 games played in a different posture. The format of the game has been described in Section 3.1. The touch data was then pre-processed as described in Section 4. In order to determine the optimal amount of training data needed to create an adequate user model, we performed the following steps:

1. The user data was split equally into train and test data. The training and testing data was extracted as described in Section 5.
2. 10 train sets of varying sizes were created by varying the amount of training data from 10%-100% of the complete training data (in steps of 10%).
3. Each of the 10 training sets was used to create a user model based on the Random Forest classifier.
4. For every user, the 10 user models were tested against the testing set (which is of a constant size).
5. The mean EER for all users at every training set size was calculated.

The variation in mean equal error rate with training set size is shown in Fig. 3. Note that once the training set size reaches 80% (data from 4 sessions), the EER of the classifier does not decrease by more than 1% upon adding more training data. Based on this analysis, we decided to collect data over 8 sessions in the full-fledged data collection in order to gather sufficient data for training and testing sets.

5.5. Determining the best performing classification algorithm

Based on the conclusions from the above mentioned pilot study, the full fledged data collection was performed. Using this complete dataset (described in Section 3.3), the next objective was to

Table 5
Mean classifier performance over all 31 users as the training set size is varied. The values in parentheses show the standard deviation.

Training Data Used %	Random Forest	SVM	Logistic Regression	Naïve Bayes	MLP
10	17.6% (7.8%)	24.1% (8.6%)	31.4% (15.0%)	23.9% (7.8%)	53.0% (17.6%)
20	12.0% (7.5%)	20.1% (8.0%)	33.7% (18.7%)	17.1% (7.5%)	57.2% (18.4%)
30	9.3% (6.8%)	19.6% (8.1%)	28.2% (12.8%)	16.0% (6.8%)	54.0% (14.7%)
40	8.7% (5.8%)	19.3% (7.6%)	35.0% (16.8%)	12.3% (5.8%)	61.0% (12.2%)
50	7.2% (5.8%)	19.0% (7.9%)	25.0% (11.8%)	11.9% (5.8%)	57.5% (13.0%)
60	6.5% (5.9%)	18.7% (7.7%)	24.9% (11.0%)	10.8% (5.9%)	61.4% (12.6%)
70	6.4% (4.9%)	18.8% (8.0%)	34.0% (14.5%)	10.6% (4.9%)	60.8% (15.5%)
80	5.9% (4.6%)	18.8% (7.9%)	34.3% (14.0%)	9.8% (4.6%)	58.4% (12.3%)
90	5.6% (4.2%)	18.9% (8.3%)	35.1% (12.8%)	8.6% (4.2%)	61.9% (11.8%)
100	5.4% (4.1%)	18.8% (7.8%)	33.7% (12.7%)	8.6% (4.1%)	60.8% (13.1%)

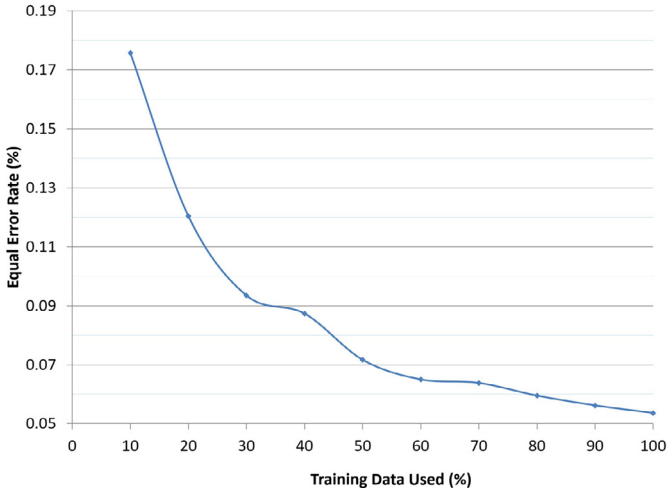

Fig. 3. Variation in EER as the size of the training set size is increased.

determine the best classification algorithm and use it as the basis for further experiments. We tested five algorithms based on a benchmark study performed by Serwadda et al. (2013). The authors here reported that Logistic Regression, SVM, Random Forest, Naive Bayes, and Multi-layer Perceptron were the best performing algorithms. To find the best classifier, we followed the same steps as described in Section 5.4. However, in this experiment, we developed five user models using each of the 10 training sets based on the five benchmark classification algorithms.

The variation in EER for each classifier as the train set size is increased is illustrated in Fig. 4. The mean EER and its standard deviation for each classifier for all users are given in Table 5. It is evident that the Random Forest classifier performed the best amongst all algorithms. When using only 10% of the total training set, it records an EER of 17.6%. As the training set size is increased, the performance increases to provide a benchmark EER of 5.4%. Comparatively, Multi-layer Perceptron was the worst performing classifier. Table 5 also indicates that Random Forest provides the most reliable classification. This is evident from the smaller change in performance as the training set size is increased. Based on these results, we decided to use Random Forest to generate user models for the remainder of the experiments.

Interestingly, some classification algorithms exhibited oscillation in the Equal Error Rate as the training set size was increased. This behavior was most pronounced for the Naive Bayes and Multilayer Perceptron algorithms. Some may suggest that this behavior is due to variation in user behavior over time and thus, a smaller training set would reflect the user's unfamiliarity with the app in the early stages. However, we believe this conclusion to be incorrect. We would like to remind the reader that the entire training set was sampled evenly in order to generate a smaller training set (in Step 2 in Section 5.4). This eliminates the effects of variation in user behavior over time. We did not pursue further investigations to determine the reason for this fluctuation in performance.

6. Experiment 1: Given a posture, does device size affect touch dynamics authentication performance?

A highly desirable characteristic of a touch based authentication system is to gather data from all devices and develop a common user model. This will either allow a user profile developed on one device to be ported to another device or merge profiles from different devices together to rapidly create a user model. This is especially useful in the now common scenario where the user owns multiple touchscreen devices, tablets and smartphones. These reasons provide the motivation to study whether the device size has any effect on the authentication performance.

In order to remove a confounding variable in this experiment, we controlled the user posture while studying the effect of device size. By user posture, we refer to user movement and the 3-D orientation of the device. We argue that in a practical system, it is possible to develop a user profile based on user postures since movement and 3-D orientation can be detected using the device's inbuilt accelerometer and gyrometer readings. To study the effect of device size, we performed a statistical comparison using the 3 Samsung built devices: T10 (10" tablet), T7 (7" tablet) and S3 (4.7" smartphone). This subset was used as it provides devices of all possible sizes available in this study while retaining manufacturer homogeneity. The test setup is illustrated in Fig. 5 and explained below:

1. For a given user and posture, a train and test set was sampled from each device using the approach described in Section 5. Based on the labels of the three devices (10: 10" tablet, *T*7: 7" tablet and *S*3: 4.7" smartphone), the train sets for any given user are labeled $Train^{T10}$, $Train^{T7}$, and $Train^{S3}$. The test sets are labeled $Test_{T10}$, $Test_{T7}$, and $Test_{S3}$.
2. Each of the three train sets is used to construct a user model using a Random Forest classifier.
3. Each of the three models was then tested on all three test sets and the Equal Error Rate was calculated in every instance. For User *i*, the EER when training on Device *X* (using $Train^X$) and testing on Device *Y* (using $Test_Y$) is labeled $EER\_i_Y^X$.
   For example, to determine the change in performance when using User *x*'s *T*10 model on another device, we restrict the posture to any one of those described in Section 3.3 and construct the user model using $Train^{T10}$ (the train set for User *x* and Device *T*10). We then test its performance on $Test_{10}$, $Test_{T7}$, and $Test_{S3}$ (the test sets for User *x* from all 3 devices). This provides the EERs $EER\_x_{T10}^{T10}$, $EER\_x_{T7}^{T10}$ and $EER\_x_{S3}^{T10}$ for that posture.

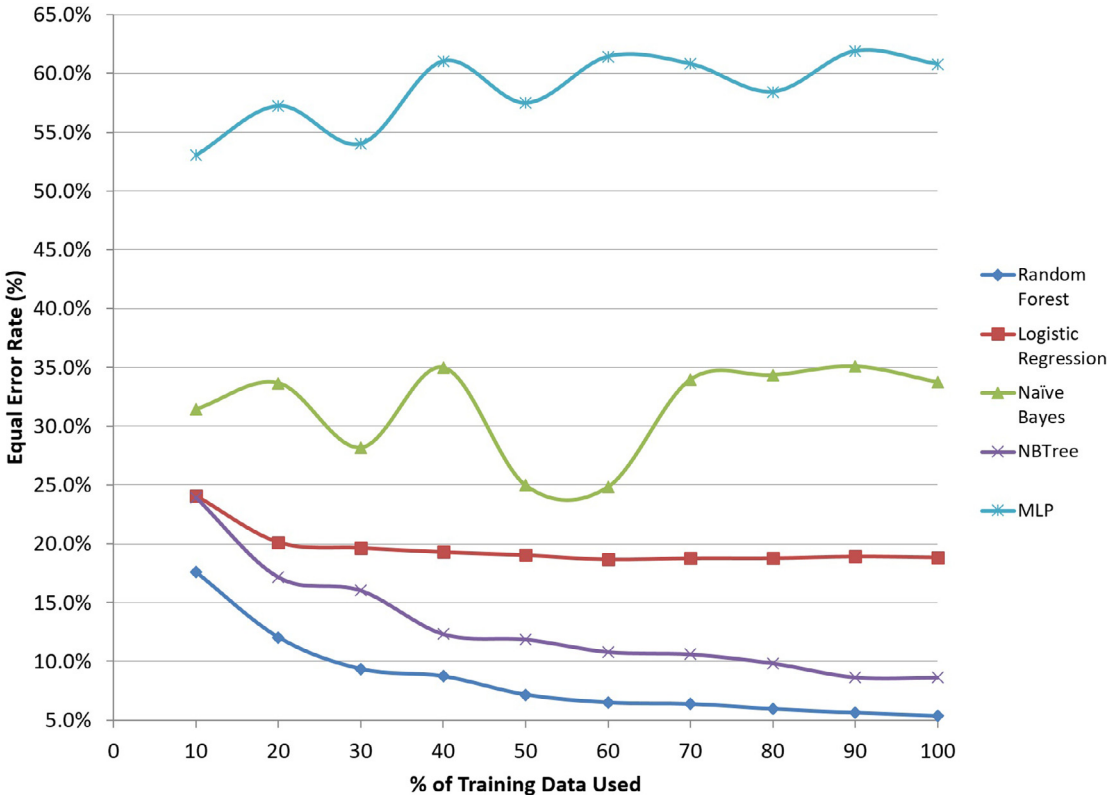

Fig. 4. Performance of various classifiers with variation in training set size. The Random Forest classifier performed the best followed by SVM.

Table 6
Test setup to measure the effect of device on authentication accuracy (given a specific posture). For a user i, a user model is created using train data from Device X ($Train^X$) and tested on Device Y ($Test_Y$). The EER for this pair is $EER\_i_Y^X$.

User	$Train^{T10}$			$Train^{T7}$			$Train^{S3}$		
	$Test_{T10}$	$Test_{T7}$	$Test_{S3}$	$Test_{T10}$	$Test_{T7}$	$Test_{S3}$	$Test_{T10}$	$Test_{T7}$	$Test_{S3}$
1	$EER\_1_{T10}^{T10}$	$EER\_1_{T7}^{T10}$	$EER\_1_{S3}^{T10}$	$EER\_1_{T10}^{T7}$	$EER\_1_{T7}^{T7}$	$EER\_1_{S3}^{T7}$	$EER\_1_{T10}^{S3}$	$EER\_1_{T7}^{S3}$	$EER\_1_{S3}^{S3}$
...	...	...	...	...	...	...	...	...	...
n	$EER\_n_{T10}^{T10}$	$EER\_n_{T7}^{T10}$	$EER\_n_{S3}^{T10}$	$EER\_n_{T10}^{T7}$	$EER\_n_{T7}^{T7}$	$EER\_n_{S3}^{T7}$	$EER\_n_{T10}^{S3}$	$EER\_n_{T7}^{S3}$	$EER\_n_{S3}^{S3}$

Table 7
List of p-values of statistical tests conducted to determine the effect of device size when the posture is controlled. The NULL hypothesis was rejected in all the tests, i.e. the EER distribution for the native device has a significantly lower mean than the other EER distribution.

Train Device:	$Train^{T10}$		$Train^{T7}$		$Train^{S3}$	
EERs compared:	EER_{T10}^{T10}-EER_{T7}^{T10}	EER_{T10}^{T10}-EER_{S3}^{T10}	EER_{T7}^{T7}-EER_{T10}^{T7}	EER_{T7}^{T7}-EER_{S3}^{T7}	EER_{S3}^{S3}-EER_{T10}^{S3}	EER_{S3}^{S3}-EER_{T7}^{S3}
Posture 1	7.94E-18	7.03E-20	1.32E-16	1.00E-16	4.37E-16	1.80E-13
Posture 2	3.08E-15	4.92E-22	3.08E-15	2.96E-12	6.31E-18	2.30E-14
Posture 3	1.05E-16	2.66E-17	4.59E-16	2.82E-13	2.83E-15	2.89E-14

When this procedure is repeated for all users, it provides a distribution of EER scores as shown in Table 6.

4. To test the effect of device size on authentication performance of a user model based on $Train^X$, the EER distributions EER_{T10}^X, EER_{T7}^X, EER_{S3}^X are compared to each other using a one-sided Student's t-test with Holm-Bonferroni correction.
5. Steps 1–4 are repeated for each of the three postures.

6.1. Student's t-test

The student's t-test is a statistical method for testing whether two normal distributions have the same mean, i.e. if the distributions are identical. A t-test may be either two-sided or one-sided. In a two-sided t-test, the null hypothesis states that the two distributions are equivalent. In a one-sided t-test, the null hypothesis states the mean of one distribution is larger or smaller than the other distribution. The null hypothesis is accepted or rejected based on the calculated test statistic t. If the observed t-statistic is more extreme than the critical value determined by the t-distribution, the null hypothesis is rejected. The critical value depends on the significance level (alpha level) of the test. This is the probability of erroneously rejecting the null hypothesis (Encyclopaedia Britannica, 2014).

In our experiments, each statistical test is a one-sided test, i.e. the p-value indicates whether one distributions of EERs is greater or less than the other. We required a significance level $\alpha = 0.05$ i.e, the p-value must be lower than 0.05 to reject the NULL hypothesis and conclude that the distributions are not equal.

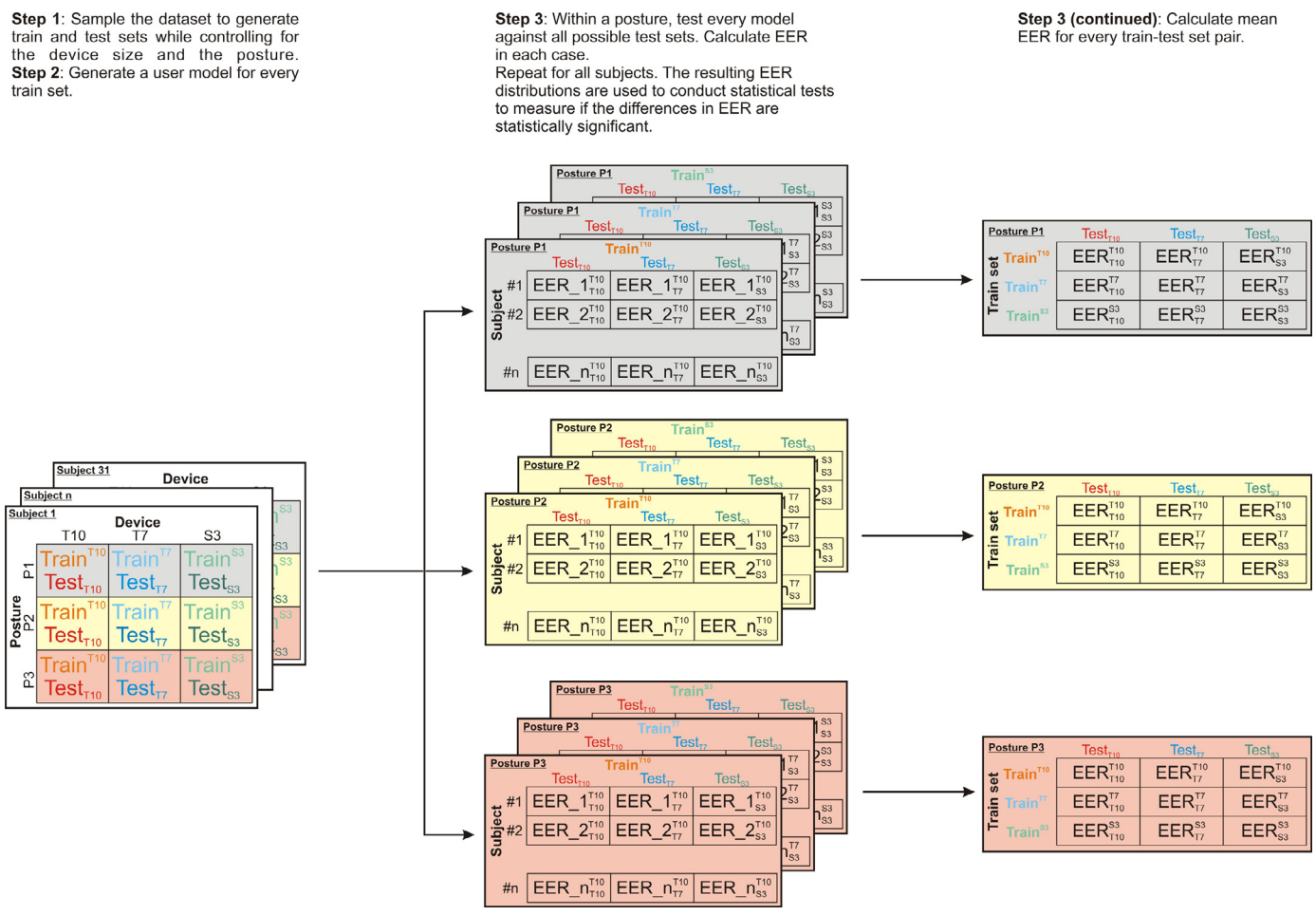

Fig. 5. The procedure to analyze the effect of device size while controlling for posture. The EER distributions in Step 3 are tabulated in the format shown in Table 6 with the results in Table 7. The final results of this analysis are shown in Table 8.

Table 8
Mean EER when the users' models are trained on one device and tested on all devices. The user posture is controlled.

Posture	$Train^{T10}$			$Train^{T7}$			$Train^{S3}$		
	$Test_{T10}$	$Test_{T7}$	$Test_{S3}$	$Test_{T10}$	$Test_{T7}$	$Test_{S3}$	$Test_{T10}$	$Test_{T7}$	$Test_{S3}$
Posture 1	5.36%	56.58%	58.69%	58.75%	6.64%	33.97%	55.02%	29.52%	8.81%
Posture 2	5.16%	53.01%	56.75%	52.33%	5.49%	34.30%	54.45%	27.51%	7.14%
Posture 3	3.80%	56.05%	54.37%	55.03%	5.42%	32.19%	51.82%	28.89%	6.60%

6.2. Holm-Bonferroni correction

Since this experiment (and the following one) employs multiple statistical tests, it is possible to reject the null hypothesis even if it is correct. This is because as the number of statistical tests increases, the probability of rejecting a null hypothesis at a given α also increases resulting in the inflation of the family-wise α level. In order to compensate for this problem, we use the Holm-Bonferroni correction for multiple comparisons. This is a sequentially rejective version of the simple Bonferroni correction for multiple comparisons and strongly controls the family-wise error rate at level α. It works as follows:

1. All p-values are sorted in order of smallest to largest, where m is the number p-values.
2. If the 1st p-value is greater than or equal to α/m, the procedure is stopped and no p-values are significant. Else, continue.
3. The 1st p-value is declared significant and now the second p-value is compared to $\alpha/(m-1)$.
4. If the 2nd p-value is greater than or equal to $\alpha/(m-1)$, the procedure is stopped and no further p-values are significant.
5. This process continues for all p-values.

6.3. Results

The outcomes of the experiment lead to the following observations:

1. The p-values from the statistical tests are shown in Table 7. The series of tests indicate that any user model performed significantly better on its native device as compared to on another device. This effect is further explained through Table 8. When a user model trained on Device X is tested on Device Y, the classifier is unable to distinguish between a genuine and impostor user. This shows that based on the current state-of-the-art techniques for feature generation, it

Table 9
List of p-values of statistical tests conducted to determine the effect of posture variation when the device type is controlled. The NULL hypothesis was rejected in all the tests, i.e. the EER distribution for the native posture has a significantly lower mean than the other EER distribution.

Train Posture:	$Train^{P1}$		$Train^{P2}$		$Train^{P3}$	
EERs compared:	EER^{P1}_{P1}-EER^{P1}_{P2}	EER^{P1}_{P1}-EER^{P1}_{P3}	EER^{P2}_{P2}-EER^{P2}_{P1}	EER^{P2}_{P2}-EER^{P2}_{P3}	EER^{P3}_{P3}-EER^{P3}_{P1}	EER^{P3}_{P3}-EER^{P3}_{P2}
T10	4.66E-06	1.24E-08	2.71E-08	1.24E-08	1.40E-10	3.03E-11
T7	8.52E-08	4.44E-10	1.23E-08	1.50E-09	2.52E-12	1.08E-10
S3	1.62E-07	8.14E-12	9.78E-10	2.76E-12	2.81E-14	6.64E-12

is not possible to use user models across devices of different sizes.
2. We note that when training and testing on data from the same device, the 10″ tablet offers the best authentication performance with an EER as low as 3.8%. Comparatively, the S3 smartphone has the worst performance with EER as high as 8.8%. This suggests that touch based authentication is more effective on devices with a larger touch screen area. This is possibly because the large screen area provides users with greater freedom to select certain screen areas to execute gestures.
3. In all cases where the model is trained and tested on the same device, the performance of our classifiers is either better or equivalent to the benchmark performances in literature. This has been achieved by using fewer features (14) than other studies in literature. For example, (Frank et al., 2013) used 27 features, (Feng et al., 2012) used 53 features and Serwadda et al. (2013) used 28 features.

7. Experiment 2: Given a device, does posture affect authentication performance?

Mobile devices offer superior portability so that users are not confined to a certain location or posture when using them. A robust touch-based authentication system must be able to authenticate users even when the user posture changes. This experiment was performed to determine the effects of changes in user posture on authentication performance. The experiment setup is identical to the previous experiment described in Section 6. However, in this case, the device size is controlled while the posture is varied to create train and test sets. Once again, only the 3 Samsung devices (T10, T7 and S3) are used in this analysis since this subset provides devices of all sizes while retaining manufacturer homogeneity.

Thus, in this experiment, for a given user and device type, 3 train sets and 3 test sets are generated: using the touch data from three postures (*P*1: device on table, *P*2: device held in portrait orientation and *P*3: device held in landscape orientation). User models are constructed using train sets $Train^{P1}$, $Train^{P2}$, and $Train^{P3}$. Each model is tested on test sets $Test_{P1}$, $Test_{P2}$, and $Test_{P3}$ and EERs generated. The statistical comparison of EER distributions is conducted in an identical fashion as the previous experiment.

7.1. Results

1. Table 9 shows that regardless of the device or posture type, the performance of a user model on its native posture is significantly better compared to when used in another posture. However, the strength of rejection is less than in the previous experiment, i.e., the *p*-values are larger. This indicates that the difference in user models caused by posture, although significant, is lesser than that caused by screen size variation.
2. The effect of posture is further explained through Table 10 where the mean EERs in each posture are shown for every device. Although Posture 1 and Posture 2 are based on the same screen orientation (portrait), the performance diminished by 2 – 4 times when a user model from one of these two postures was used in the other posture.
3. The drop in performance is even more severe (up to 10 fold) when the train posture is Posture 3. This is because Postures 1 and 2 are based on portrait orientation while Posture 3 is based on landscape orientation. The user interface changes drastically between these two orientations. In order to mitigate this problem, a possible area of future research would be to refer to gesture coordinates in terms of its proximity to certain UI elements, and not in absolute X and Y coordinates.
4. Lastly, a common trend in all scenarios was that Posture 1 (device lying flat on table) had the worst performance. Contrastingly, Posture 3 (device held in landscape mode) offered the best performance.
5. Based on this analysis, we believe that utilizing the 3-D orientation of a device may help in developing a more accurate user-specific profile as it would help to detect the user posture.

8. Experiment 3: Given a posture and device size, does device configuration affect authentication performance?

A modern smartphone or tablet consists of a variety of subsystems and hardware components. Device manufacturers typically source individual components from Original Equipment Manufacturers (OEMs) instead of building everything in-house. In the context of touch authentication, the hardware components that most affect the acquired touch profile are the digitizer (that captures touch gestures), screen size, and the software that regulates the sampling frequency and pressure readings on the device. The effect of each of these components cannot be studied in isolation. Thus, we treat the entire package as one *device configuration*.

The experiments in the previous two sections show that both posture and device size have a significant effect on a touch based authentication system. We also noted that the T7 and S3, being closer in size, displayed better performance when their user models were cross-tested. Thus, our next step was to determine the effect of device configuration and if a user model is portable between similar sized devices. To do so, the experiment was setup exactly as in Section 6. However, only two devices were used in this experiment: the HTC EVO and the Samsung S3. This is because both devices have the same display size (4.7") and similar hardware configuration. The test setup is illustrated in Table 11. To reiterate:

1. On every device, for a given user and posture, a pair of training and testing sets were extracted using the interleaved approach. The training and testing sets contained the same number of strokes.
2. The training/testing sets were extracted using the touch data from two devices (S3 and EVO). The training sets for a user are labeled $Train^{S3}$, and $Train^{EVO}$. The testing sets are labeled $Test_{S3}$, and $Test_{EVO}$.

Table 10
Mean EER when the users' models are trained on one posture and tested on all postures. The device size is controlled.

Device	$Train^{P1}$			$Train^{P2}$			$Train^{P3}$		
	$Test_{P1}$	$Test_{P2}$	$Test_{P3}$	$Test_{P1}$	$Test_{P2}$	$Test_{P3}$	$Test_{P1}$	$Test_{P2}$	$Test_{P3}$
T10	5.36%	21.11%	36.83%	20.78%	5.16%	23.94%	41.84%	30.14%	3.80%
T7	6.64%	23.31%	34.00%	21.37%	5.49%	26.31%	35.75%	27.45%	5.42%
S3	8.81%	19.67%	33.65%	21.64%	7.14%	37.02%	39.86%	37.05%	6.60%

Table 11
Test setup to measure the effect of device configuration on authentication accuracy.

User	$Train^{S3}$		$Train^{EVO}$	
	$Test_{S3}$	$Test_{EVO}$	$Test_{S3}$	$Test_{EVO}$
1	$EER\_1_{S3}^{S3}$	$EER\_1_{EVO}^{S3}$	$EER\_1_{S3}^{EVO}$	$EER\_1_{EVO}^{EVO}$
...	...	...	...	...
n	$EER\_n_{S3}^{S3}$	$EER\_n_{EVO}^{S3}$	$EER\_n_{S3}^{EVO}$	$EER\_n_{EVO}^{EVO}$

Table 12
List of p-values of statistical tests conducted to determine the effect of device configuration when the posture is controlled. The data shows that there is a strong difference between touch profiles based on the device configuration.

Train Device: *EERs compared:*	$Train^{S3}$ $Test_{S3}$-$Test_{EVO}$	$Train^{EVO}$ $Test_{EVO}$-$Test_{S3}$
Posture 1	2.51E-08	1.72E-10
Posture 2	7.79E-08	2.01E-11
Posture 3	2.86E-10	2.84E-09

3. Two user models were constructed using $Train^{S3}$, and $Train^{EVO}$.
4. Each model was then tested separately on $Test_{EVO}$ and $Test_{S3}$ and the Equal Error Rate was calculated in every instance. The EER when training on Device *X* and testing on Device *Y* for User *i* is labeled $EER\_i_Y^X$.
   For example, in order to determine the effect of using the EVO model on the S3 for User *x*: We restrict the posture and construct the user model for User *x* using the train set for User *x* from EVO ($Train^{EVO}$) and then benchmark on the test sets for User *x* from EVO and S3 ($Test_{EVO}$ and $Test_{S3}$). This provides the EERs $EER\_x_{EVO}^{T10}$ and $EER\_x_{S3}^{T10}$ for that posture.
   When this procedure is performed over all users, it provides a distribution of EER scores as shown in Table 11.
5. To test the effect of device configuration on authentication performance of a user model based on Device *X*, the EER distributions EER_{S3}^X and EER_{EVO}^X are compared to each other using Student's *t*-test with Holm-Bonferroni correction.

8.1. Results

1. Table 12 shows the statistical results of the experiments. As Table 12 shows, the performance of a user model trained on one device is significantly different from when that same profile is used on another device.
2. This effect is explained in Table 13 where the mean EERs on each device for this experiment are shown. When a user model trained on Device *X* is used on a test set from Device *Y*, the error rate roughly doubles in each case. This happens even though the device size and posture is the same. It should be noted that the drop in performance is far less compared to Experiment 1 where a different sized device from the same manufacturer was used.

Table 13
Mean EER (over all users) when the model is trained on one device and tested on the other device. Note that the EER worsens by 100% even though the devices share technical and physical specifications.

	Posture 1		Posture 2		Posture 3	
	$Test_{S3}$	$Test_{EVO}$	$Test_{S3}$	$Test_{EVO}$	$Test_{S3}$	$Test_{EVO}$
$Train^{S3}$	8.8%	16.5%	7.1%	14.6%	6.6%	14.0%
$Train^{EVO}$	17.5%	8.1%	17.4%	6.7%	16.7%	5.5%

3. Based on the observations in Section 6 and from this experiment, it is evident that a similar device size results in a similar user profile. In this experiment, since the manufacturers of the devices are different, this may have led to a larger drop in EER when the user model is ported over. The exact reasons for variation in performance forms an interesting area for further research.
4. When training and testing on data from the same device, the S3 offers better authentication performance compared to the EVO. This may be due to the hardware and/or software configuration of the devices. However, determining the exact play of variables that cause this performance difference is infeasible because a device configuration is a combination of hardware and software components that function together to create its unique behavior. Thus, testing each component independently is not possible.
5. We note that the trend from Experiment 2 is exhibited here too: Posture 1 provides the worst authentication performance, while Posture 3 provides the best performance for the EVO and S3 smartphones.

9. Effect of inter-device sensor differences when transforming features to port user models across devices

Since this study deals with porting user models to devices of different sizes for training and testing purposes, it may seem natural to normalize features across all devices. For example, the start and stop coordinates of gestures, gesture length, and the pressure readings may seem to be features amenable to such transformations. In this section, we discuss technical hurdles related to such transformations.

9.1. Transforming X-Y coordinates and related features

There are two approaches to transforming coordinate-based features:

1. Pixel-dimension based scaling
2. Physical-dimension based scaling

9.1.1. Pixel-dimension based scaling

This technique scales the pixel length and pixel coordinates of gestures. It uses the ratio of pixel dimensions of the source and target devices as a scaling factor.

We will use devices *T*10 and *S*3 to illustrate this point. Both have similar display resolution but different physical dimensions

(see Table 1). If the objective is to use pixel-dimension based scaling, no feature scaling is needed since the pixel width and height of both devices is similar. This is precisely the case of the results shown in Table 8 for the ($Train^{T10}$, $Test_{S3}$) and ($Train^{S3}$, $Test_{T10}$) pairs. As we clearly see from the EER values for these train-test pairs, this technique leads to a significant degradation in performance. We can thus infer that this scaling technique is not beneficial in cases where the device sizes are significantly different. This is because a user's gestures do not scale linearly in the physical dimension based on the device screen size.

In case the issues in feature scaling was caused by using a linear scale, we attempted using both neural-net based prediction and clustering-based prediction to transform the X-Y coordinates of the train set. However, neither technique was successful at accurately predicting gesture coordinates on the test set. Our analysis indicated that the user's touch profile does not change in a generalizable pattern when the device size is varied (when measured in terms of X-Y coordinates).

We also theorize this technique may not work even if two devices are of the same size. For example, if a device has double the pixel resolution as another device of the same physical size, the user models would need to be scaled so they are similar in size in the *physical dimension*. However, as we explain next, even this optimization is insufficient when porting user models.

9.1.2. Physical-dimension based scaling

This technique also scales the gestures pixel length and coordinates. However, it does so based on the ratio of the physical dimensions of the source and target devices.

The simplest scenario of such scaling occurs when two devices are of identical physical size and screen resolution. No scaling is required in such a scenario. In fact, it is reasonable to conclude that the generated user models must be indistinguishable from each other.

This should be the case with the *S*3 and *EVO* devices as they share the same physical and pixel specifications (see Table 1). In such a case, we expect the performance difference to be negligible when training and testing using ($Train^{EVO}$, $Test_{S3}$) and ($Train^{S3}$, $Test_{EVO}$) pairs. However, as Table 13 shows, even this simple scaling scenario doubles the inaccuracy of the model as measured through EER rates.

9.2. Transforming pressure data

With regards to the pressure readings, it is important to remember that the presure readings on capactive touchscreens are inferred indirectly based on the screen occlusion by the finger. Since skin is deformable, it will occlude a greater screen area under higher pressure. Capacitive touchscreen-based devices leverage this fact to sense the pressure applied. However, due to this indirect sensing mechanism, each device uses a different scale to measure pressure. One device may provide a much more granular pressure reading, while another may report the pressure using only a few pre-defined values. Likewise, some devices may report the same pressure reading for all pressure values beyond an arbitrary threshold.

We encountered these issues when attempting to scale pressure values across all devices. The variation in pressure readings meant that there was no effective method to normalize pressure readings across all devices. We therefore chose to not scale the pressure readings across devices.

9.3. Summary

Based on the above analysis, we can infer the following:

1. No X-Y coordinate scaling technique that relies on the ratio of the devices' physical dimensions can improve performance. This is because of all the features used in our user models (see Table 4), only two are unrelated to gesture coordinates (StartPressure and StopPressure) and their information gain contribution is minimal.
2. We can also infer that the difference in user models is caused by differences in the accuracy of the two devices in collecting touch data. This difference in performance is an inherent side-effect of porting user models to other devices (of a different configuration).
3. Pressure readings are currently a highly inaccurate feature when transforming user models across devices of different configuration.
4. Using devices with similar sizes but different configurations (such as phones from different manufacturers) with the assumption that the data collected will be equivalent is a critical error. Based on our experiments, we believe this may lead to artificially increased accuracy rates. This is because the inter-device differences in reading touch information acts as a confounding factor when attempting to leverage inter-user differences in user models. This threat to validity is likely to be present in the datasets released by Frank et al. (2013) and Antal et al. (2015) that use devices with similar specifications but from different manufacturers. We caution researchers to evaluate the equivalence of user models across devices with similar specifications before using them together in empirical experiments.
5. Whether X-Y coordinates-based or pressure-based features are accurate across identical devices is an important research topic that must be investigated further. Such a study would determine if the differences in user models are due to relative differences in sensors of devices with different specifications, sensor noise, or a combination of both. If sensor noise is a significant factor, then it may lead to inflated authentication performance in datasets that used identical devices for the purposes of the data collection (such as Serwadda et al. (2013) and Mahbub et al. (2016)). This would happen since inter-device sensor differences would magnify differences between user profiles. Researchers should treat this as a threat to the validity of conclusions until evidence to the contrary emerges.

10. Time to authentication

A continual authentication system actively monitors user behavior to detect impostors. Due to the unconstrained nature of this authentication mechanism, there may be a lull in user activity, preventing the system from rapidly determining the user's identity. Due to this, a continual authentication system requires more time to authenticate a user as compared to static authentication measures. Thus, the time to authentication is an important performance metric for continual authentication systems.

Table 14 illustrates the procedure to calculate the time to authentication for the touch-based authentication system proposed in Section 5. Note that our system requires 5 strokes to generate a decision. Based on this information and statistics from the collected dataset, our authentication system is able to determine the user's identity in 18 seconds with the EER performance summarized in Table 15.

Please note that this performance and time-to-authentication is based upon the frequency with which users submitted strokes during the data collection. Due to the focused nature of the data collection where the users interacted with the device without a pause, 18 seconds is possibly the shortest possible time-to-authentication for our system. In practical situations, the user may

Table 14
Statistics on an average user in the collected dataset and expected time to authentication. Please note that the provided statistics are for any given device-posture combination.

# of devices	4
# of postures per device	3
Total number of games (per user-device-posture)	8
Avg. time taken to complete a game	3 *mins.*
Avg. total time spent per device-posture	24 *mins.*
Total strokes recorded	1252
# of strokes executed/min.	$\frac{1252}{8\times 9} = 17$
Strokes reqd. for authentication	5
Time to authentication	$\frac{5}{17} \times 60 = 18$ *seconds*
% of strokes lost in noise removal	13%
Strokes/user in pruned dataset	$\left(\frac{13080}{4\times 3}\right) = 1090$
Horizontal:Vertical strokes	5: 1
# of horizontal strokes	908
# of vertical strokes	182

Table 15
Mean EER for our touch-based authentication system when using the native device-posture user models. (**Posture 1**: Device on table; **Posture 2**: Device held in portrait mode; **Posture 3**: Device held in landscape mode).

Device	Posture		
	P1	*P2*	*P3*
T10	5.36%	5.16%	3.80%
T7	6.64%	5.49%	5.42%
S3	8.81%	7.14%	6.60%

be less focused and may execute gestures with reduced frequency leading to a longer time to authentication. This problem can be offset by reducing the number of strokes used to make a decision. This would, however, impact the accuracy of authentication.

It is likely that the system's authentication performance is proportional to the number of strokes used to make a decision about the user's identity. For example, using ten strokes to authenticate a person will probably reduce the EER further, but increase the time-to-authentication. This trade-off between strokes-to-decision and time-to-decision is an interesting research topic that merits further investigation.

11. Critical discussion

11.1. Controlling confounding factors

The experimental setup allowed us to control certain confounding factors:

1. Variations due to the type of device used as the manufacturer, model, touch screen technology (capacitive, resistive), screen size, screen resolution, aspect ratio, and touch screen sampling rate. This was controlled by using the same set of devices for all test subjects.
2. Variations caused by user movement during device interaction. This was controlled by conducting the data collection in a laboratory environment with the same seating arrangement for all test subjects.
3. Variations in user behavior due to extended device use. The users were expected to complete one session per sitting. The reason for this was to extract a user behavior over a period of time. At the same time, users would have learnt to use the app as they familiarized with it. However, this behavior was part of the study and is not considered a confounding factor.

Some confounding factors in this study are:

1. **Repeated testing** - The users habituated to the app over time. However, this was done intentionally in order to incorporate information on user habituation into the data. Hence, it is not a threat to its validity.
2. **Impostor Data** - Due to the limited number of test subjects in our study, we cannot generate a sufficiently varied impostor model. In the real world, impostor touch data could vary dramatically. Only a study with a larger user base would allow us to build reasonable impostor models.
3. **Targeted attack versus random attack** - While other test subjects function as impostors for a given subject, they did not try to mimic the touch behavior of the user. A more sophisticated impostor would watch the user for some time and mimic his/her behavior. However, as a counterpoint, it seems improbable that an impostor could learn to mimic the amount of pressure and of acceleration at different parts of the stroke just by watching the genuine user.
4. **Interaction of selection and treatments** - All test subjects in this data collection were students with ages ranging from 16 to 30. This demographic is not representative of the general population. Furthermore, only one subject indicated that he/she used a mobile touch screen device weekly. Three subjects indicated that they had rarely or never used a touch screen device before. Once again, this is not representative of the general population. However, it should be noted that an continual authentication system based on touch dynamics will primarily be used by people who own such a device. In this case, it is preferable that the dataset only contain users who are used to a touch screen device so that we obtain realistic results.
5. **Reactive arrangements** - The test subjects were aware of the purpose of the data collection. This may impact the results of this study.
6. **Effect of user specific characteristics** - Gender, dominant hand, mental state may have an effect on the user behavior. While the gender and dominant hand characteristics were collected as part of the dataset, we did not use it in the data analysis.
7. **Environmental variables -** While the environment was controlled, users were free to come in at any time of the data to complete their sessions. The time of day and/or user mood may have had an effect on the touch data collected.

11.2. Miscellaneous details on the experimental setup

1. During the data collection, a maximum of four test subject submitted data concurrently. In all cases, the order in which the devices were used was randomized. All subjects used the same devices in a seated position in a laboratory setting. The subjects were isolated to prevent imitation.
2. The same set of devices was used for all test subjects to minimize the effect of device variation. The devices were factory reset to their default settings before the data collection to eliminate the impact of installed apps and software aging. All devices ran the same version of the Android operating system.
3. The same set of pictures was used on all devices but the order of images was randomized after every game.
4. The data submission process typically took 1–2 weeks. This format was used in order to incorporate information on user acclimatization and the effect of device usage over multiple sessions. We have analyzed the effect of acclimatization in this dataset and published it in another work (Palaskar et al., 2016).

5. Data was collected for both device orientations only when the device was held (Postures *P2* and *P3*). We excluded the "On Table-LandscapeOrientation" posture that pairs with Posture *P1* ("On Table-PortraitOrientation" described in Section 3.3). This was primarily to minimize session length and loss of subjects due to onset of boredom. However, the effect of screen orientation can be measured with Postures *P2* and *P3* and extrapolated to other scenarios.
6. The app was designed such that the UI elements scaled according to the screen resolution and aspect ratio, but stayed in the same position relative to each other, i.e. we used a 'relative layout' format. Thus, any variation in user profile when using different device sizes are not due to differences in the UI layout, but only due to the effect of a different screen size.
7. We chose to use simple horizontal and vertical strokes in our data collection as the aim of the study was to study the effect of posture and device size. Complex gestures such as multi-finger swipes would have interfered with the experimental objectives.

11.3. Extending the results to other usage scenarios

Our app simulates a particular usage scenario where the user searches for a particular image. A number of researchers have shown that touch data is application and context dependent (Feng et al., 2014; Shen et al., 2016b; Khan et al., 2014). Thus, it is possible that our results do not extend to other apps and activities such as web browsing, e-mail, watching video, etc.

11.4. Influence of sample size

We performed the data collection over a period of 2 weeks per subject. We finally used 31 subjects in this this experimental analysis. This sample size is sufficient for the statistical tests that we are conducting. However, a real-world setting has a significantly large number of users and impostors and its effect on our technique's performance cannot be determined except with a much larger study. New versions of Android OS do not allow implementing a 'background' touch-based authentication system. Thus, experiments on touch based authentication can currently only be performed using simulated apps and environments. This severely limits the ability of practitioners to develop robust touch-based authentication systems using large datasets.

11.5. Data density per test subject

While the data was collected from multiple test subjects, some subjects used far more strokes to complete a game. In order to elicit the most uniform results, we equalized the number of strokes used per subject to build a model. To do so, each subject's dataset was pruned such that the number of strokes per subject was the same as that of the subject with the lowest number of strokes. While this gives us the benefit of a uniform classifier, it must be noted that additional strokes are available for most users and were not used in the data analysis.

12. Summary

In this work, we investigated the underlying dynamics of touch based authentication. Specifically, we analyzed the effect of device size, user posture and device configuration on a touch-based authentication system's performance using a locally collected dataset. Various parameters such as user environment, devices used, session length, etc. were controlled in order to remove confounding factors.

- Our approach resulted in a classifier performance that is either better or equivalent to the benchmark performances reported in literature.
- This benchmark performance has been achieved using 14 features (explained in Section 4.2). This is fewer features than other studies in literature. For example, Frank et al. (2013) used 27 features, Feng et al. (2012) used 53 features and Serwadda et al. (2013) used 28 features.
- We demonstrated that the user posture, device size, and the device configuration have a significant effect on the classifier performance, with the device size having the greatest effect and the device configuration the least. To develop resilient touch based authentication systems, it would be worthwhile to sense the user posture via the 3-D orientation readings from the device's gyrometer and incorporate the readings into user models.
- Our results indicate that a larger device size leads to better authentication rates. At the same time, current state-of-the-art attributes cannot authenticate across changes in device size or screen orientation.
- Scaling techniques to port user models from one device to another suffer due to the inherent noise and inaccuracy of the devices. Due to this, even the simplest form of porting (using user models interchangeably across similar devices) leads to significant degradation in authentication performance. As a corollary, using similar devices during data collection while assuming that they are interchangeable is an unrecognized threat to the validity of the resultant empirical conclusions.

Our work provides a rigorous empirical analysis of the effects of three important factors that impact the performance of a touch-based authentication system: user posture, device size and device configuration. These contributions allow practitioners to recognize the limitations of the current state-of-the-art gesture-based authentication algorithms and develop novel attributes that lead to more robust touch-based authentication systems.

We have also made the dataset publicly available to advance the technology in this domain.

13. Future work

Based on our findings, we are currently developing novel touch dynamics based features that are resilient to changes in environmental variables such as user posture, device size and configuration. Furthermore, there are other factors, such as the effect of manufacturer, variation amongst devices with similar sizes, and stroke sampling methods. These factors may also affect a touch-based authentication system and need to be researched further. We are currently extending the dataset to study the effect of these variables on authentication performance. The current dataset is being analyzed to examine features that are more sensitive to the device size and user posture.

Touch-based authentication is a relatively new field with considerable scope for further research. The dataset created during the course of this work will be made publicly available in the near future. Within and without this dataset, there are a number of research areas that need to be investigated. Some of these are described below:

1. Effect of User Interface design on performance: As Section 6 showed, there is a significant drop in performance when a user model is ported to a larger or smaller device. We believe one of the major reasons for this is that the UI expands and contracts with screen size. This in turn affects the coordinates of the touch (such as start and stop points). Since this data forms one of the key features of the user model, it is natural for user models to be incompatible between devices of significantly different screen sizes.

Instead of referring to a gesture's location by its absolute coordinates, referring to its location with reference to UI elements may be a more natural means to determine user behavior. We theorize that the location where the users tend to execute gestures is dependent not on the physical screen device, but rather on the UI. For example, during our data collection, we observed that users prefer to touch the photos themselves and not on their side, even when they changed the screen orientation (and thus the screen aspect ratio) in different postures.

2. An analysis of the dataset revealed that each device employed its own scheme for representing pressure. Some devices used a scale of 0.00 to 1.00, while others functioned within a much narrower scale. While this functions as identifying characteristic for different devices, it also causes issues when a user model needs to be ported. One area of future research would be to normalize pressure readings according to the destination device. This would increase compatibility between different devices.
3. While each test subject took part in the same number of sessions, the number of strokes submitted by each user varied. This is because some users chose to use shorter strokes when navigating, thus employing more strokes to finish their session. The effect of the number of strokes was removed from this study in order to remove a confounding factor. However, the typical length of a user's stroke and the number of strokes used to complete a task would form a useful feature that could be incorporated in a user's touch model. This topic requires further investigation.
4. While our dataset consists of both horizontal and vertical strokes stored separately, our classifier was trained on a combination of both. Other researchers have taken the approach of constructing separate classifiers for each stroke type (Serwadda et al., 2013). This strategy may lead to better performance as it simplifies the data supplied to each classifier.
5. As described in Section 5, sequential groups of strokes were used in generating a training set. In our experiments, each group contained 5 strokes. It is possible that increasing the group size in the training set will lead to more reliable performance. Similarly, increasing the group size in the testing set may lead to better results at the expense of greater time to authentication. The trade-off between these variables is an interesting area for further research.

References

Android Studio, 2013. Get Android Studio. https://developer.android.com/studio/

Antal, M., Bokor, Z., Szabo, L.Z., 2015. Information revealed from scrolling interactions on mobile devices. Pattern Recognit. Lett. 56, 7–13. doi:10.1016/j.patrec.2015.01.011.

Cai, L., Chen, H., 2011. Touchlogger: inferring keystrokes on touch screen from smartphone motion. HotSec 11, 9.

Encyclopaedia Britannica Online Academic Edition, 2014. Student's *t*-test. http://www.britannica.com/EBchecked/topic/569907/Students-t-test

Federal Communications Commission, 2012. Announcement of new initiatives to combat smartphone and data theft. https://www.fcc.gov/document/announcement-new-initiatives-combat-smartphone-and-data-theft

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. IEEE, pp. 451–456.

Feng, T., Yang, J., Yan, Z., Tapia, E.M., Shi, W., 2014. Tips: Context-aware implicit user identification using touch screen in uncontrolled environments. In: Proceedings of the 15th Workshop on Mobile Computing Systems and Applications. ACM, New York, NY, USA, pp. 9:1–9:6. doi:10.1145/2565585.2565592.

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.

Gartner, 2013. Market share analysis: Mobile phones, worldwide, q2 2013.

IDC, 2017. Smartphone os market share, 2017 q1.

Jain, A., Kanhangad, V., 2015. Exploring orientation and accelerometer sensor data for personal authentication in smartphones using touchscreen gestures. Pattern. Recognit. Lett. 68 (Part 2), 351–360. doi:10.1016/j.patrec.2015.07.004. Special Issue on Soft Biometrics.

Khan, H., Atwater, A., Hengartner, U., 2014. Itus: An implicit authentication framework for android. In: Proceedings of the 20th Annual International Conference on Mobile Computing and Networking. ACM, New York, NY, USA, pp. 507–518. doi:10.1145/2639108.2639141.

Li, L., Zhao, X., Xue, G., 2013. Unobservable reauthentication for smart phones. In: Proceedings of the 20th Network and Distributed System Security Symposium, NDSS, 13.

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

Lu, L., Liu, Y., 2015. Safeguard: user reauthentication on smartphones via behavioral biometrics. IEEE Trans. Comput. Social Syst. 2 (3), 53–64.

Mahbub, U., Sarkar, S., Patel, V.M., Chellappa, R., 2016. Active user authentication for smartphones: a challenge data set and benchmark results. CoRR abs/1610.07930.

Miguel-Hurtado, O., Stevenage, S.V., Bevan, C., Guest, R., 2016. Predicting sex as a soft-biometrics from device interaction swipe gestures. Pattern. Recognit. Lett. 79, 44–51.

Mondal, S., Bours, P., 2015. Continuous authentication and identification for mobile devices: combining security and forensics. In: Information Forensics and Security (WIFS), 2015 IEEE International Workshop on. IEEE, pp. 1–6.

Mondal, S., Bours, P., 2015. Swipe gesture based continuous authentication for mobile devices. In: Biometrics (ICB), 2015 International Conference on. IEEE, pp. 458–465.

Nader, J., Alsadoon, A., Prasad, P., Singh, A., Elchouemi, A., 2015. Designing touch-based hybrid authentication method for smartphones. Procedia Comput. Sci. 70, 198–204.

Nixon, K.W., Chen, X., Mao, Z.-H., Chen, Y., 2016. Slowmo-enhancing mobile gesture-based authentication schemes via sampling rate optimization. In: Design Automation Conference (ASP-DAC), 2016 21st Asia and South Pacific. IEEE, pp. 462–467.

Nohara, T., Uda, R., 2016. Personal identification by flick input using self-organizing maps with acceleration sensor and gyroscope. In: Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication. ACM, New York, NY, USA, pp. 58:1–58:6. doi:10.1145/2857546.2857605.

Palaskar, N., Syed, Z., Banerjee, S., Tang, C., 2016. Empirical techniques to detect and mitigate the effects of irrevocably evolving user profiles in touch-based authentication systems. In: 2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE), pp. 9–16. doi:10.1109/HASE.2016.39.

Pew Research Center, 2017. Cell phone and smartphone ownership demographics. http://www.pewinternet.org/data-trend/mobile/cell-phone-and-smartphone-ownership-demographics/

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

Serwadda, A., Phoha, V.V., Wang, Z., 2013. Which verifiers work?: A benchmark evaluation of touch-based authentication algorithms. In: Biometrics: Theory, Applications and Systems (BTAS), 2013 IEEE Sixth International Conference on. IEEE, pp. 1–8.

Shen, C., Zhang, Y., Cai, Z., Yu, T., Guan, X., 2015. Touch-interaction behavior for continuous user authentication on smartphones. In: 2015 International Conference on Biometrics (ICB), pp. 157–162. doi:10.1109/ICB.2015.7139046.

Shen, C., Zhang, Y., Guan, X., Maxion, R.A., 2016. Performance analysis of touch-interaction behavior for active smartphone authentication. IEEE Trans. Inf. Forensics Secur. 11 (3), 498–513.

Shen, C., Zhang, Y., Guan, X., Maxion, R.A., 2016. Performance analysis of touch-interaction behavior for active smartphone authentication. IEEE Trans. Inf. Forensics Secur. 11 (3), 498–513. doi:10.1109/TIFS.2015.2503258.

Shih, D.H., Lu, C.M., Shih, M.H., 2015. A flick biometric authentication mechanism on mobile devices. In: 2015 International Conference on Informative and Cybernetics for Computational Social Systems (ICCSS), pp. 31–33. doi:10.1109/ICCSS.2015.7281144.

Sitová, Z., Šeděnka, J., Yang, Q., Peng, G., Zhou, G., Gasti, P., Balagani, K.S., 2016. Hmog: new behavioral biometric features for continuous authentication of smartphone users. IEEE Trans. Inf. Forensics Secur. 11 (5), 877–892.

Syed, Z., Helmick, J., Banerjee, S., Cukic, B., 2015. Effect of user posture and device size on the performance of touch-based authentication systems. In: 2015 IEEE 16th International Symposium on High Assurance Systems Engineering, pp. 10–17. doi:10.1109/HASE.2015.10.

Xu, H., Zhou, Y., Lyu, M.R., 2014. Towards continuous and passive authentication via touch biometrics: an experimental study on smartphones. In: Symposium On Usable Privacy and Security, SOUPS, 14, pp. 187–198.

Xu, H., Zhou, Y., Lyu, M.R., 2014. Towards continuous and passive authentication via touch biometrics: an experimental study on smartphones. In: Symposium On Usable Privacy and Security (SOUPS 2014). USENIX Association, Menlo Park, CA, pp. 187–198.

Zhang, H., Patel, V.M., Fathy, M., Chellappa, R., 2015. Touch gesture-based active user authentication using dictionaries. In: Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on. IEEE, pp. 207–214.

Zahid Syed received his Masters (2008) and PhD (2014) from West Virginia University. He is currently an Assistant Professor at the University of Michigan - Flint. He previously worked as an ASIC design engineer and as a software engineer before pursuing a PhD and academia. He is the recipient of a Best Paper Award at the 2011 IEEE International Symposium on HASE. His research interests include behavioral-biometrics based authentication systems, user privacy & security in relation to computer systems and social networking, and developing pedagogical tools for teaching Computer Science courses through an active learning environment.

Jordan Helmick is a statistician currently working in the healthcare industry. In 2013 he received a Master's Degree in Statistics from West Virginia University. Additionally, Jordan is an independent researcher with interests in the application and theory of spatial statistics, experimental design, and statistical consulting. He especially enjoys multidisciplinary collaborations that allow him to combine statistical and domain-specific knowledge to solve research problems more efficiently.

Dr. Sean K. Banerjee is an Assistant Professor in the Department of Computer Science at Clarkson University in Potsdam, NY where he is the co-director and co-founder of the Terascale All-sensing Research Studio. Prior to coming to Clarkson, Sean was a post-doctoral researcher at Carnegie Mellon University in Pittsburgh, PA. He received his Ph.D from West Virginia University in 2014. His research lies at the intersection of machine learning, natural language processing and software engineering. He was co-author on the HASE 2011 Best Paper titled "Effects of user habituation in keystroke dynamics on password security policy".

Dr. Bojan Cukic is a Professor and Chair of the Department of Computer Science at the University of North Carolina at Charlotte, where he also serves as the Executive Director of the Data Science Initiative. His research interests include software engineering with emphasis on verification and validation, resilient computing, biometrics and computer science education. Dr. Cukic served as a program or general chair of several IEEE symposia and conferences, including 2018 International Symposium on Software Reliability Engineering (ISSRE). He received MS and PhD degrees in Computer Science from the University of Houston.