Gesture-Based Continuous Authentication for Wearable Devices: The Smart Glasses Use Case

Jagmohan Chauhan[1,2(✉)], Hassan Jameel Asghar[2], Anirban Mahanti[2], and Mohamed Ali Kaafar[2]

[1] UNSW, Sydney, Australia
[2] Data61, CSIRO, Sydney, Australia
{jagmohan.chauhan,hassan.asghar,anirban.mahanti, dali.kaafar}@data61.csiro.au

Abstract. We study the feasibility of touch gesture behavioural biometrics for implicit authentication of users on smart glasses by proposing a continuous authentication system on Google Glass using two classifiers: SVM with RBF kernel, and a new classifier based on Chebyshev's concentration inequality. Based on data collected from 30 users, we show that such authentication is feasible both in terms of classification accuracy and computational load on Glass. We achieve a classification accuracy of up to 99 % with only 75 training samples using behavioural biometric data from four different types of touch gestures. To show that our system can be generalized, we test its performance on touch data from smartphones and found the accuracy to be similar to Glass. Finally, our experiments on the permanence of gestures show that the negative impact of changing user behaviour with time on classification accuracy can be best alleviated by periodically replacing older training samples with new randomly chosen samples.

1 Introduction

Since many wearable devices store highly sensitive user information such as health data, a secure and usable authentication mechanism to restrict access to unauthorized users is paramount. A straightforward solution is *entry-point* authentication relying on personal identification numbers (PINs), passwords or graphical patterns [18]. However, frequent use of entry-point authentication potentially disrupts user activities [2,4]. Moreover, in comparison to smartphones, unlocking patterns on wearable devices such as Google Glass are more vulnerable to *shoulder-surfing* [14,19] since the Glass touchpad is easily observable from a distance.

An alternative is to use an implicit and continuous authentication system, which runs in the background without disrupting the user, and authenticates

The full version of this paper is available at http://arxiv.org/abs/1412.2855.

M. Manulis et al. (Eds.): ACNS 2016, LNCS 9696, pp. 648–665, 2016.
DOI: 10.1007/978-3-319-39555-5_35

the user whenever he/she performs a designated action, which in our case is using the touchpad. The system only triggers entry-point authentication if an intrusion is detected. Provided the method is reliable, this approach reduces the number of times a legitimate user needs to undergo entry-point authentication. Many continuous authentication schemes have previously been proposed in the literature for smartphones [7,9,10,19], however, they may not provide similar accuracy or may be computationally heavy on wearables such as Glass. A smaller touchpad of Glass compared to a smartphone is likely to show less variation in gestures across different users, thereby impacting accuracy. Also, running computationally expensive applications can deplete the battery faster on Glass [11].

These factors motivate a feasibility study of continuous authentication on wearables. Towards this goal, in this paper, we assess the feasibility of continuous authentication on Glass. Our key contributions are as follows. First, to the best of our knowledge, we are the first to study the feasibility of touch gesture based continuous authentication on smart glasses in terms of classification accuracy and computational cost by using Google Glass as a use case. Although Glass itself may or may not be continued as a product, our work is still relevant since our scheme can be extended to other smart glasses with touchpads namely RECON, SiME, GlassUP, ORA-S and Icis, as well as other touchpad devices, e.g., smartphones. Second, we model a touch gesture as one or more *forces* applied on the touchpad by the user's finger over the duration of the gesture. A resulting novel feature is the *downward force feature* which is a product of pressure and size values extracted from the device's touch event.

Third, to authenticate the user, besides using support vector machine (SVM) with Gaussian radial basis function (RBF) classifier (widely used for continuous authentication on smartphones), we introduce a new classifier based on *Chebyshev's concentration inequality*. Previous research on touch gesture based continuous authentication on smartphones has shown that during testing (authentication), instead of using features from a single sample of a gesture, using features from a *block* of samples of the gesture shows improved classification accuracy [7,10,15]. We note that this observation implicitly uses the assumption that the average value of a feature over a block is more likely to be concentrated around the mean. The justification of this comes from concentration inequalities, which give probabilistic bounds on the deviation of the average of identically distributed random variables from their true mean. This led us to propose the Chebyshev classifier. Lastly, by extending our experiments to smartphone touch data, we find that the size of the touchpad has an effect on classification accuracy; smaller touchpads, as in smart glasses, exhibit less variation across users.

2 Related Work

Entry Point Authentication: Zheng et al. [20] collected the tapping behaviour of 80 different users when entering PINs on smartphones and extracted four

features (acceleration, pressure, size, and time) from the collected data, achieving a 3.65 % equal error rate. Shahzad et al. [16] created a system named GEAT. However, unlike the scheme proposed by Zheng et al., GEAT differentiates the user on the basis of their sliding behaviour and uses unique features such as finger velocity, device acceleration, and slide time, achieves a 0.5 % equal error rate. Similarly, Luca et al. [6] exploit user sliding behaviour while unlocking smartphone patterns, achieving an accuracy of around 50 %. In comparison to the these works, our study focuses on continuous authentication.

Continuous Authentication: Numerous schemes [3,7,10,19] have been proposed for continuous authentication on smartphones. Hui et al. [19] collected data from 31 volunteers for different touch operations such as keystroke, slide, pinch and handwriting to test their continuous authentication scheme and showed that the slide gesture is the best in classifying users, while handwriting performs the worst. Similarly, Frank et al. [7] proposed a scheme using a set of 30 touch-based features and tested it on 41 users. Their classifier achieved a median equal error rate of 0 % within the same usage session and 2–3 % across different sessions. The reason why these two schemes achieve exceptionally high authentication accuracy might be due to the fact that users were static and were given specific tasks to be performed. In comparison, we did not enforce any such restriction on the users. Li et al. tested a continuous authentication scheme based on sliding and tap gestures [10] and extracted features such as the position and area of first touch, duration and average curvature of slide. SilentSense [3] used finger movements and user motion patterns and achieved 99 % accuracy. In contrast to our study, the temporal effect of user behaviour on accuracy is not studied in the last two schemes. A more recent work from Mondal and Bours [12] uses a trust-based approach for continuous authentication, where instead of waiting for a fixed number of gestures from the user before making a decision, the system updates its trust value, about the current user being the target user, with every gesture and locks the user when the trust value falls below a pre-defined threshold. This approach can be applied to any continuous authentication mechanism including ours.

A somewhat related topic is the recently introduced sensor-enhanced keystroke dynamics [8], which augments traditional timing-based keystroke dynamics with motion sensors available on smart devices. Not only does this approach increase the accuracy of traditional keystroke dynamics and gesture-based authentication [8], it has also been shown to be more resistant to statistical attacks using general population statistics [17].

Overall our work is different from previous works in three major ways: (1) we assess the feasibility of touch gestured based continuous authentication on smart glasses. Smart glasses, such as Google Glass, present unique challenges such as smaller form factor and lesser computational power compared to smartphones, (2) we propose a new classifier based on concentration inequalities, and (3) we propose new force-based features.

3 Background

The Google Glass: Google Glasses (cf. Fig. 1a) contain an optical display mounted on the lens, which contains a small screen (cf. Fig. 1b). The user can navigate using voice commands or by interacting with the touchpad located on the side through taps or swipes (forward, backward or downward). Swipes can be done through one, two or three fingers. Note that not all apps (cards) and their menu items can be interacted using voice and require a touch gesture.

(a) Frame (b) Display

Fig. 1. The Google glass (images courtesy of Wikipedia and Google).

Definitions: For the rest of this paper, a *gesture* is defined as a tap or a swipe with one finger on the touchpad. For each gesture, the set of data recorded by the Glass touchpad, e.g., the point of contact, is called a *sample*. A sample contains a time-ordered sequence of one or more *readings*, which correspond to data recorded at different discrete time intervals during the duration of a gesture. Each reading contains data corresponding to one or more variables called *features*. The authentication mechanism takes as input a set of gestures and either (implicitly) accepts or rejects the user depending on whether or not the set matches the gestures of the target user. True positive rate (TPR) is defined as the fraction of times the target user is correctly accepted. False positive rate (FPR) is defined as the fraction of times the attacker is (wrongly) accepted as the target user. Equal error rate (EER) is defined as the rate at which both acceptance and rejection *errors* are equal, i.e., when $1 - \text{TPR} = \text{FPR}$. EER is widely used as a measure of classification accuracy. A related measure is the average error rate (AER), which is defined as $\frac{1}{2}(1 - \text{TPR} + \text{FPR})$ and is useful when EER is unknown. Receiver operating characteristic (ROC) curve shows the trend of TPR against FPR. Variability in these rates is introduced by changing different parameter values of the authentication system.

4 Continuous Authentication for Google Glass

4.1 Architecture

The proposed system architecture, as shown in Fig. 2, has a training and a testing phase. The system listens for gesture events that are triggered whenever the user performs gestures on the touchpad. Once an event is triggered, elementary features such as the start and end point of gestures are extracted. From the start and end points, the gesture type (tap, forward, backward or downward

swipe) is identified, after which higher-level features, e.g., force exerted on the touchpad, are derived. Some of the features in our system are derived as a function of time and require further processing for consistent inter-comparison. After going through this post-processor, our system feeds the resulting features to the classifier. During training, the classifier generates different classification models for different gesture combinations. During the testing phase, real-time gesture data from the current user is processed to obtain the feature sets as above, which are then fed to the classifier for prediction.

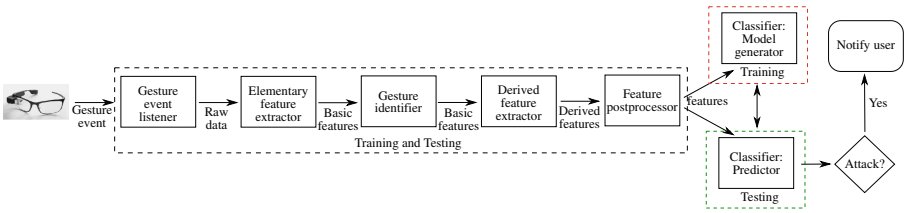


Fig. 2. System architecture.

4.2 Data Collection

We collected data for four gestures: tap, forward swipe, backward swipe, and downward swipe from the Glass touchpad (v 18.1, Android) using a background process, which reads the raw touch data values at runtime. Glass is equipped with the Synaptics T1320 touchpad. More technical details, such as the structure of touch packets, are given in the full paper. We selected 30 volunteers consisting of 8 females and 22 males within the 18–45 age bracket and asked them to use Google Glass for a few hours. All were colleagues and students with a computer science background. They were free to explore Glass as they liked and use any application installed on the device. Each user was trained how to operate Glass prior to data collection. Table 1 shows the quantity of gesture data collected from the users. Forward swipe is the most frequently used gesture, followed by the tap; downward swipe being the least frequent gesture. Backward swipes can be used in place of forward swipes to navigate in the opposite direction, explaining their relatively less usage. Moreover, downward swipes are mostly used for quitting an app or cancelling an action and hence their frequency is the lowest.

Table 1. Total number of samples, average and minimum sample size per user, and average gap (in seconds) for gestures obtained in our user study.

Gesture	Total	Ave. sample size	Min. sample size	Average gap
Tap (T)	4932	164.4	60	13
Forward swipe (F)	7874	262.46	67	8
Backward swipe (B)	3257	108.56	37	17
Downward swipe (D)	1525	50.83	11	32

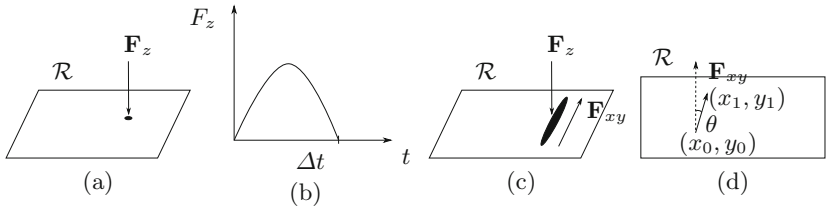


Fig. 3. Force based gesture models: (a) the tap force, (b) the magnitude of the force curve $F_z(t)$ over the interval Δt, (c) the two forces active during a swipe, and (d) the source of the force $\mathbf{F}_{xy}$ estimated through the angle θ.

4.3 Gesture Model and Feature Extraction

We model the touchpad as a rectangle $\mathcal{R}$ on a two dimensional xy-plane, where the origin is the bottom-left corner. We distinguish between two types of gestures, tap (T) and swipe. Swipe is further divided into forward (F), backward (B) and downward (D). We model each gesture as one or more forces (exerted by user's finger) acting over the course of a gesture. Our main assumption is that the *magnitude and source of these forces over the time duration* of the gesture are characteristics of a user.

Modelling the Tap Gesture: The tap is characterised by the downward force applied by the finger on the touchpad. This force, denoted $\mathbf{F}_z$, acts downwards on $\mathcal{R}$, i.e., along the z-axis. The source is the point on $\mathcal{R}$ where the user taps. This is shown in Fig. 3a. The magnitude of $\mathbf{F}_z$ is calculated using pressure P and area (size) A readings from the touch event as $F_z = PA$. Note that our hypothesis is that it is the correlation between the pressure and area values that is expected to be consistent across samples, instead of treating the two separately, as is done in [6] for instance. As the tap is performed over a time interval, say Δt, we denote the magnitude of $\mathbf{F}_z$ over time as $F_z(t)$, which is a *time series*. Figure 3b visualises the possible shape of F_z over the duration of tap. $F_z(t)$ can be calculated over discrete points t in the interval Δt through corresponding pressure and area values. We also use tap duration (Δt) as a feature.

Modelling the Swipe Gesture: We model a swipe as two forces acting on $\mathcal{R}$ simultaneously. The first is $\mathbf{F}_z$, the force acting downwards on $\mathcal{R}$, as in the case of tap. The second, denoted $\mathbf{F}_{xy}$, is a force acting along the direction of swipe (xy-plane). These two forces are visualized in Fig. 3c. To estimate the source of $\mathbf{F}_z$, we use the start point (x_0, y_0) and the end point (x_1, y_1) of the swipe. The source of the force $\mathbf{F}_{xy}$ is estimated as the angle θ between the straight line joining these two points and the y-axis as shown in Fig. 3d. To estimate the duration of the forces, in addition to the swipe duration Δt, we also include the swipe length l. The magnitude of $\mathbf{F}_z$ is again estimated as the time series $F_z(t)$ of individual pressure and area (PA) values. The magnitude of $\mathbf{F}_{xy}$ is also modelled as a time series $F_{xy}(t)$ with the difference that individual values are the magnitude of *velocity* at discrete time intervals. This is done since in classical mechanics, force

Table 2. List of features.

Gesture	#	Feature	Symbol	Gesture	#	Feature	Symbol
Tap	1	tap x-coordinate	x	Swipe	1	start pt. x-coordinate	x_0
	2	tap y-coordinate	y		2	start pt. y-coordinate	y_0
	3	down. force time series	$F_z(t)$		3	end pt. x-coordinate	x_1
	4	tap duration	Δt		4	end pt. y-coordinate	y_1
					5	angle	θ
					6	down force time series	$F_z(t)$
					7	planar force time series	$F_{xy}(t)$
					8	swipe duration	Δt
					9	swipe length	l

is considered proportional to acceleration which can be determined by change in velocity. Table 2 summarizes the list of features.

Post-processing the Time Series: The time series for the magnitude of force ($F_z(t)$ and $F_{xy}(t)$) can be misaligned due to the non-uniform sampling rate of the device and difference in duration of the gesture. To get a consistent comparison of time series from different readings, we do the following: (a) we align the first sample of the two time series at time $t = 0$; (b) we resample each time series at intervals of $t_{\mathsf{int}} = 0.01\,\mathrm{s}$ (slightly lower than the system average of $\approx 0.012\,\mathrm{s}$) similar to the approach is used in [16]; (c) we use a cut-off point $t_{\mathsf{off}} = 0.3$, after which all values are discarded. Most time series span an interval Δt, which is less than t_{off}. For such cases, all values at time $\Delta t < t < t_{\mathsf{off}}$ are mapped to 0.

4.4 Chebyshev Classifier

Many researchers have indicated that a *block* of samples used for testing shows an improved performance over using individual samples [7,10,15], where the average reading of the feature over the block is used as a single instance for input to the classifier. We note that if a sample block is to be used, a classifier based on *concentration inequalities* can be employed. A concentration inequality bounds the probability that a random variable deviates from its expected value. The deviation from the expected value decreases (probabilistically) with an increase in the block size of identically distributed random variables. We thus propose a one class classifier based on the concentration inequality called Chebyshev's inequality. The use of this inequality is not unprecedented in anomaly or outlier detection in a somewhat different manner [1]. A further advantage of Chebyshev's inequality is that it does not make any assumptions on the probability distribution of data (which may be unimodal or multimodal).

Let X be a random variable representing a unitary feature, i.e., any feature other than a time-series based feature. Let $\mathbf{x} = (x_1, \ldots, x_n)$ denote n samples of this unitary feature. The corresponding random variables are denoted $X_1, \ldots, X_n$. We assume that these random variables are independent and identically distributed (i.i.d.), since they correspond to different samples (of the same gesture type). Let $\mathrm{E}[X] = \mu_X$ and $\mathrm{Var}[X] = \sigma_X^2$ denote the expected value (mean) and

variance of X, respectively. Then for any $\tau > 0$, $\Pr\left[|X - \mathrm{E}[X]| \geq \tau\right] \leq \frac{\mathrm{Var}[X]}{\tau^2} \Rightarrow \Pr\left[|X - \mu_X| \geq \tau\right] \leq \frac{\sigma_X^2}{\tau^2}$ is known as Chebyshev's inequality [13, Sect. 8, p. 431]. Consider the random variable $\overline{S}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$. Since the X_i's are i.i.d., we have $\mathrm{E}[\overline{S}_n] = \frac{1}{n}\sum_{i=1}^{n}\mathrm{E}[X_i] = \frac{n}{n}\mu_X = \mu_X$, and $\mathrm{Var}[\overline{S}_n] = \mathrm{Var}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] = \frac{1}{n^2}\mathrm{Var}\left[\sum_{i=1}^{n} X_i\right] = \frac{1}{n^2}\sum_{i=1}^{n}\mathrm{Var}[X_i] = \frac{n}{n^2}\sigma_X^2 = \frac{\sigma_X^2}{n}$. Using Chebyshev's inequality on $\overline{S}_n$ and the above two results, we get

$$\Pr\left[\left|\overline{S}_n - \mathrm{E}[\overline{S}_n]\right| \geq \tau\right] \leq \frac{\mathrm{Var}[\overline{S}_n]}{\tau^2} \Rightarrow \Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu_X\right| \geq \tau\right] \leq \frac{\sigma_X^2}{n\tau^2} \quad (1)$$

for any $\tau > 0$. A qualitative explanation of this inequality is that as n increases, the average of a sample is more likely to be concentrated around the mean. Now, let $\rho = \frac{\sigma_X^2}{n\tau^2}$. Rearranging we get $\tau = \frac{\sigma_X}{\sqrt{n\rho}}$. By specifying a value of ρ in this equation, i.e., a bound on probability, we can obtain a corresponding threshold τ. This then gives us a straightforward classification method for features: Given a sample $x'_1, x'_2, \ldots, x'_n$, purported to be generated from the same distribution as X, we calculate the sample mean and see if this lies within the threshold τ determined by ρ. If yes, then the sample is classified as belonging to the target user; otherwise it is rejected. Similarly, for a time-series based feature we can use this classifier with slight modification as detailed in the full version of the paper. Thus given an n-element sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and the parameter ρ, we have the *Chebyshev feature classifier* $f(\mathbf{x}, \rho)$ which outputs 1 if the sample belongs to the target user and 0 otherwise. To make an overall decision given samples from a set of m features $\chi = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, we have the following classifier, which we call the *Chebyshev classifier*:

$$g(\chi, \rho, \epsilon) = \begin{cases} 1, & \text{if } \sum_{i=1}^{m} f(\mathbf{x}_i, \rho) > \epsilon m \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

We call ϵ the *decision threshold* and ϵm the *decision boundary*. Through our experiments we found $\epsilon = \frac{2}{3}$ to give the best EER.

4.5 SVM Classifier

Our second classifier is the binary class SVM with Gaussian radial basis function (RBF) kernel. We used its implementation available through the LIBSVM library [5].To construct the feature space for SVM, we represented the time series based features as $\frac{t_{\mathsf{off}}}{t_{\mathsf{int}}} = 30$ dimensional vectors. The whole feature space of the SVM is then a vector of all unitary features and time series based features represented in the aforementioned way. Constructed in this way, the SVM classifier is given training data. To obtain the best classification results, we performed a grid search with 10-fold cross validation on the training data to find the optimal values for its parameters, i.e., C and γ [5]. Notice that the training phase needs data both from the legitimate (target) user and other users (represented as the second class). As this type of data represents unbalanced data (more data from

the second class), we used a weighted scheme SVM. After a user model has been created by the SVM, the authentication phase or testing phase can be carried out. Let χ be a set of samples of features to be tested against the user model, where we assume the sample size of each feature to be $n \geq 1$. For each feature $\mathbf{x} \in \chi$ with n samples denoted by $\mathbf{x} = (x_1, \ldots, x_n)$, the average value $\frac{1}{n}\sum_{i=1}^{n} x_i$ is used in the final feature vector.

5 Evaluation and Results

5.1 Experimental Setup

To evaluate the performance of Chebyshev classifier, we consider three sets of users denoted by U_1, U_2, and U_3, containing 10, 20 and 30 users, respectively. For all user sets, our experimental setup is as follows. To obtain the True Positive Rate (TPR), we randomly select a target user, and use a random set of 50 samples from this user as the training set. The test set used for authentication, consists of the remaining samples. Given a fixed value of n, a random sample of length n is obtained from the test set. The random test sample is then fed to the classifier, which was trained using the training data. The decision from the classifier is then logged. This process was repeated 500 times each with a new random target user. Note that due to randomness, the training set for the same user is different over different trials. Finally, the number of times, out of the 500 tests, the target user was accepted was used to compute TPR.

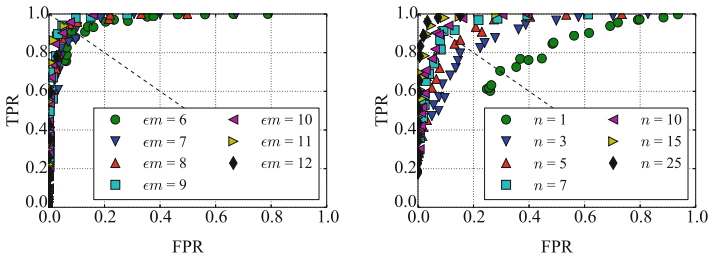
The False Positive Rate (FPR) is calculated in the same manner as TPR except that the classifier was given a test sample of size n from all the samples of a random attacker selected from U_1 (respectively from U_2, and U_3), excluding the target user. FPR was calculated as the rate at which the attacker was accepted. The size of the training set for tap and forward swipe was 50, whereas backward swipe and downward swipe had training set sizes of 25 and 10, respectively, since for these gestures we had lower number of available samples (see Table 1).

For the SVM classifier we divided the pool of 30 users into three disjoint sets. The first set, labelled U_1, consists of 10 target users for whom we had *at least* 75 samples for all gesture types and is fixed. The remaining 20 users are modelled as attackers and are assigned to two sets labelled U_2 (10 attackers) and U_3 (20 attackers). For each user in U_1, the training data consists of a random sample of a fixed size from the user's data. This constitutes positive samples for the target user required for binary class SVM training. The negative samples for the target user came from the data of the remaining 9 target users in U_1. That is, the data from the remaining 9 users was used in the training phase to model the *mock* attacker. The data of the users from U_2 and U_3 is used to compute FPR.

5.2 Chebyshev Classifier Results

First, we empirically determined the decision threshold ϵ in Eq. 2. For this, we used the user set U_1, and chose tap and forward swipes as gestures. Since tap

and forward swipes have a total of $m = 13$ features (cf. Table 2), ϵm ranges from 6 (majority decision) to 12 (unanimous decision). We construct a ROC curve for each of these cases. As n increases we observe that majority decision does not produce the best result. Figure 4a shows the ROC curves when $n = 15$. The different values of FPR and TPR are obtained by varying the probability parameter ρ in the Chebyshev classifier from 1.00 to 0.1 with steps of 0.05. The dashed line in the figure is the line with TPR $= 1 -$ FPR, which meets the ROC curve at the EER value.



(a) Different values of $\lceil \epsilon m \rceil$ (b) Different values of n

Fig. 4. ROC curves - Chebyshev classifier.

We can see no significant improvement beyond $m = 9$. Since $\epsilon m = 9$ implies $\epsilon \approx 0.69$, we use the nearest approximation $\epsilon = \frac{2}{3}$ and the decision boundary $\lceil \epsilon m \rceil$ for the Chebyshev classifier in Eq. 2. This corresponds to the two-third majority rule. Table 3 shows the decision boundaries for various combination of gestures used in our evaluation which are obtained by choosing $\epsilon = \frac{2}{3}$.

Table 3. The decision boundaries corresponding to the decision threshold $\epsilon = \frac{2}{3}$ for different combination of gestures from the Chebyshev classifier.

Combination	$\lceil \epsilon m \rceil$	m	Combination	$\lceil \epsilon m \rceil$	m
T	3	4	T + F + B	15	22
F/B/D	6	9	T + F + B + D	21	31
T + F	9	13			

Next, we studied the impact of n on the EER. Figure 4b shows the EER for the combination T + F against different values of n with the user set U_1 (notice that there are n taps and n forward-swipes in each test sample). The ROC curves show improvement as n increases, starting with an EER of about 30 % for $n = 1$ and an EER of around 3 % for $n = 25$. The trend of improving EER with increasing n is shown by all gesture combinations and all user sets, U_1, U_2 and U_3, as shown in Table 4. Note that for a gesture combination containing multiple gestures, e.g., T + F, authentication can trigger as soon as it collects a *minimum* of n samples for each gesture. From Table 4, we observe that the tap

gesture as a standalone gesture performs worse in terms of EER as compared to the swipes. The EER of the forward and backward swipes are comparable, with forward swipes narrowly edging out. The downward swipe performs worse than the other two swipe types, which is potentially due to fewer data points available for training. The EER deteriorates by 3 to 4 percent when using the data sets U_2 (20 users) and U_3 (30 users) as compared to data set U_1 (10 users). However, we do not see a noticeable deterioration in EER when comparing data sets U_2 and U_3, which suggests that adding more number of users to the system does not deteriorate the accuracy of the system by a huge factor. Our most important gesture combination is T + F since the bulk of activities on Glass can be performed by a combination of these two gestures. With $n = 10$ taps and forward swipes each, EER is less than 10 %.

Table 4. EER for different gesture combinations and n - Chebyshev classifier (Glass).

Combination	Set	n							Set	n					Set	n				
		1	3	5	7	10	15	25		1	3	5	7	10		1	3	5	7	10
T	U_1	0.35	0.27	0.23	0.21	0.18	0.16	0.13	U_2	0.38	0.32	0.25	0.23	0.19	U_3	0.37	0.29	0.25	0.22	0.20
F		0.32	0.23	0.15	0.14	0.12	0.07	0.07		0.35	0.23	0.18	0.16	0.13		0.33	0.25	0.18	0.18	0.14
B		0.32	0.22	0.17	0.14	0.12	-	-		0.34	0.26	0.21	0.18	0.16		0.36	0.28	0.24	0.22	0.19
D		0.33	0.26	0.20	0.19	0.17	-	-		0.32	0.26	0.20	0.18	0.17		0.34	0.23	0.20	0.20	0.14
T + F		0.29	0.18	0.14	0.09	0.09	0.05	0.03		0.33	0.21	0.16	0.13	0.12		0.32	0.20	0.18	0.14	0.10
T + F + B		0.27	0.16	0.09	0.08	0.04	-	-		0.30	0.17	0.13	0.11	0.07		0.30	0.22	0.15	0.10	0.07
T + F + B + D		0.25	0.13	0.09	0.07	0.03	-	-		0.27	0.13	0.11	0.07	0.06		0.26	0.16	0.09	0.07	0.06

Finally we also looked at the relationship of EER with ρ, and found that for a given n and gesture combination a fixed value of ρ can be used which appears independent of the size of the user set. Details are in the full version of the paper.

5.3 SVM Classification Results

The accuracy of the SVM classifier as measured by the average error rate (AER) is shown in Table 5. The classification accuracy is varied against two parameters: training size $|T|$ and testing size n for each gesture combination listed in the table. The training set size was varied from 25 to 75 at intervals of 25. Note that AER for all gesture combinations decreases with increasing training size, since it gives the classification algorithm more information for accurate prediction. However, this may also lead to *overfitting*, which is indeed the case with downward swipe with training set of size 75. The AER of the SVM classifier also improves with increasing number of test samples, i.e., n. The tap gesture performs the worst amongst all the individual gestures and forward swipe outperforms all other gestures, which is consistent with the observation from the Chebyshev classifier. As observed with Chebyshev classifier earlier, the AER does not significantly deteriorate with more number of users in the system (U_3 against U_2).

5.4 Distinguishing Features

To determine if individual features have distinguishing capabilities, we use the Chebyshev feature classifier f on user set U_2 to obtain true positive (TP) and

Table 5. AER for different gesture combinations and n - SVM classifier (Glass).

Combination	Training Size	Set	n					Set	n				
			1	3	5	7	10		1	3	5	7	10
T	25	U_2	0.40	0.32	0.30	031	0.26	U_3	0.37	0.34	0.32	0.29	0.30
	50		0.30	0.32	0.28	0.29	0.30		0.36	0.31	0.30	0.26	0.27
	75		0.30	0.29	0.27	0.28	0.27		0.32	0.31	0.30	0.28	0.27
F	25		0.32	0.25	0.20	0.20	0.19		0.31	0.26	0.21	0.19	0.18
	50		0.27	0.21	0.21	0.22	0.20		0.26	0.21	0.19	0.18	0.18
	75		0.28	0.21	0.18	0.19	0.15		0.23	0.22	0.19	0.18	0.16
B	25		0.33	0.32	0.31	0.31	0.30		0.33	0.35	0.29	0.31	0.29
	50		0.28	0.27	0.26	0.27	0.23		0.32	0.29	0.26	0.23	0.21
	75		0.29	0.27	0.27	0.25	0.21		0.31	0.28	0.25	0.24	0.21
D	25		0.33	0.27	0.23	0.20	0.16		0.34	0.26	0.20	0.19	0.16
	50		0.30	0.21	0.18	0.16	0.17		0.30	0.22	0.17	0.16	0.14
	75		0.30	0.28	0.27	0.30	0.32		0.31	0.28	0.29	0.29	0.29
T + F	25		0.35	0.24	0.20	0.18	0.17		0.32	0.23	0.19	0.19	0.18
	50		0.30	0.21	0.18	0.16	0.17		0.29	0.21	0.17	0.17	0.15
	75		0.26	0.17	0.12	0.11	0.11		0.30	0.13	0.12	0.11	0.10
T + F + B	25		0.28	0.20	0.16	0.14	0.14		0.29	0.21	0.18	0.16	0.15
	50		0.29	0.14	0.11	0.10	0.07		0.27	0.14	0.10	0.08	0.06
	75		0.23	0.12	0.10	0.10	0.09		0.20	0.14	0.10	0.09	0.08
T + F + B + D	25		0.25	0.18	0.16	0.13	0.12		0.28	0.17	0.16	0.15	0.15
	50		0.21	0.09	0.06	0.04	0.03		0.22	0.12	0.09	0.08	0.07
	75		0.15	0.08	0.04	0.03	0.01		0.16	0.09	0.06	0.05	0.03

false positive (FP) frequencies for the features of all four gestures as shown in Fig. 5. The x-axis shows 31 features (4 for tap plus 9 each for forward, backward and downward swipes). The TP frequencies are above 400 (out of 500) for all gesture types except the downward swipe (last nine features in the figure), which is most likely due to its small training set size, i.e., 10. Nevertheless, observe that the FP frequencies are lower than the corresponding TP frequencies for all features. We therefore included all features for classification as each can effectively distinguish between users. For more details of the setup and exact frequencies, see the full version of the paper.

5.5 Comparison of the Two Classifiers

To compare the two classifiers in terms of classification accuracy, we use EER readings from the Chebyshev classifier based on the set of 20 users, i.e., the set U_2 shown in Table 4, and we use the AER readings from SVM based on training set of size 50 from Table 5.[1] We first consider $n = 10$ for the purpose of our

[1] Note that when $1 - \text{TPR} = \text{FPR}$ (as is the case with EER), AER and EER are the same and hence comparable.

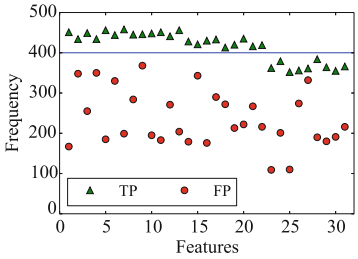


Fig. 5. TP & FP frequencies obtained via Chebyshev feature classifier for all features.

comparison. By looking at Tables 4 and 5 we can see that compared to the SVM classifier, Chebyshev's error rate is lower for taps, forward swipes and backward swipes. For all other combinations the two classifiers have similar error rates. For other values of n, we observe that the SVM classifier performs slightly better when $n = 1$, but the Chebyshev classifier's performance rapidly improves with increasing n, outperforming SVM in the three aforementioned gesture types. For combination of gestures, the performance of the two is very similar. These findings suggest that in terms of accuracy both classifiers are effective on Glass and hence can be used on similar wearables.

To compare the computational overhead of the two classifiers, we evaluated the time taken by model generation and prediction. Both these components are illustrated in Fig. 2. We first implemented both components of the two classifiers on a desktop computer. The SVM classifier was implemented in Java (via LIBSVM), whereas we used Python to implement the Chebyshev classifier. The results of the model generation and prediction time are shown in Table 6.

Table 6. Model generation and prediction time (ms) for gestures on a PC.

	Tap	Swipe		Tap	Swipe
Chebyshev Model	11	20	Chebyshev Predictor	0.04	0.095
SVM Model	38,000	49,000	SVM Predictor	9	9.4

Not surprisingly, for both the classifiers model generation takes longer than prediction. For both model generation and prediction, the Chebyshev classifier is many orders of magnitude faster than SVM. This suggests that using SVM for training on Glass can be computationally expensive in terms of power and heat generation. However, three important points need to be considered here. First, high model generation time is not inherent to SVM. In fact, it is due to the use of the RBF kernel; a linear SVM is likely to yield much lower model generation time. Secondly, we do not consider the high model generation time as a drawback of the SVM classifier, as (a) model generation is done infrequently, and (b) model generation can be outsourced to the Cloud (depending on connectivity). Lastly, a smaller grid search, i.e., restricting the ranges of the parameters C and γ,

may result in faster model generation time, at the possible expense of accuracy. Alternatively, although the optimum range of these SVM parameters depend on user data, it may be possible to experimentally determine whether the optimum values lie within narrow ranges for touch based gestures. Nevertheless, our focus was more on accuracy than speed.

We, therefore, chose to implement only the predictor component of SVM on Glass to check the actual performance. The classification models were generated offline on a desktop computer and loaded on to the Glass. On the other hand, for Chebyshev classifier we implemented both the model generator and predictor on Glass. The results from our experiment are shown in Table 7. As can be seen, Chebyshev is faster than SVM in terms of prediction time and needs little time for model generation on Glass. Having said that, the prediction time for SVM is also small enough to be practical. In terms of space requirements, both classifiers require storing gesture data which is in the order of a few kilobytes. For the model, Chebyshev classifier needs to store the means, variances and co-variances for all features, whereas the SVM classifier needs to store the support vectors. The model space complexity also increases with gesture combinations. Typically, the model size ranges from 15 KB for a simple tap to 400 KB for all gestures. In any case, Glass has 8 GB of storage capacity, and the total space required by the classifiers is only in the order of a few megabytes. The main advantage of using the Chebyshev classifier, in our opinion, is its ease of implementation (as it requires standard functions and therefore does not require external libraries).

Table 7. Model generation and prediction time (ms) for different gestures on Glass.

	T	F	T + F	T + F + B	T + F + B + D
Chebyshev Model	150	325	499	838	1,172
Chebyshev Predictor	0.80	0.32	1.13	1.89	2.74
SVM Predictor	24	40	70	90	110

5.6 Generalization: Results on Smartphone Data

To test the generalizability of our proposed system on smartphones, we used publicly available smartphone gesture data which was collected by the authors of [19].[2] The data consists of 120 taps, and 20 forward, backward and downward-swipes each for 31 users. We chose 30 of the 31 users for our study. We further fixed training size of 50 for taps and 10 for all swipe gestures. The rest of the data was used as the testing set. The other details of the experimental setup remain the same as in Sect. 5.1. The results of applying Chebyshev and SVM on the smartphone data are shown in Tables 8 and 9, respectively.

The trends observed in the results for both the classifiers on the smartphone data remain similar to Glass data. We observe that the accuracy of the system

[2] The data is available from http://xuhui.me/.

Table 8. EER for different gesture combinations and n - Chebyshev classifier (phone).

Combination	Set	n					Set	n					Set	n				
		1	3	5	7	10		1	3	5	7	10		1	3	5	7	10
T	U_1	0.43	0.30	0.24	0.23	0.15	U_2	0.39	0.27	0.21	0.18	0.15	U_3	0.36	0.26	0.22	0.17	0.16
F		0.17	0.05	0.06	0.03	0.03		0.16	0.07	0.07	0.06	0.04		0.16	0.10	0.07	0.07	0.04
B		0.20	0.13	0.12	0.11	0.09		0.16	0.12	0.10	0.11	0.10		0.22	0.15	0.11	0.10	0.11
D		0.20	0.11	0.08	0.06	0.06		0.21	0.13	0.09	0.08	0.07		0.18	0.10	0.09	0.08	0.07
T + F		0.16	0.09	0.06	0.04	0.03		0.16	0.08	0.05	0.05	0.04		0.16	0.08	0.06	0.04	0.05
T + F + B		0.15	0.05	0.02	0.03	0.03		0.12	0.04	0.02	0.02	0.02		0.12	0.05	0.04	0.03	0.02
T + F + B + D		0.09	0.03	0.01	0.02	0.01		0.08	0.03	0.02	0.01	0.01		0.09	0.03	0.02	0.01	0.02

Table 9. AER for different gesture combinations and n - SVM classifier (phone).

Combination	Set	n					Set	n				
		1	3	5	7	10		1	3	5	7	10
T	U_2	0.43	0.41	0.38	036	0.35	U_3	0.44	0.40	0.38	0.38	0.36
F		0.12	0.06	0.05	0.04	0.04		0.10	0.05	0.04	0.04	0.04
B		0.21	0.14	0.12	0.11	0.11		0.19	0.16	0.15	0.11	0.10
D		0.14	0.08	0.07	0.05	0.03		0.12	0.08	0.06	0.05	0.05
T + F		0.28	0.19	0.10	0.05	0.03		0.27	0.19	0.08	0.05	0.03
T + F + B		0.19	0.10	0.06	0.04	0.03		0.20	0.10	0.05	0.04	0.03
T + F + B + D		0.15	0.10	0.04	0.03	0.02		0.20	0.10	0.04	0.04	0.02

increases with increasing testing size, i.e., n. The system is able to achieve accuracy of 98 %-99 % with $n \geq 7$ with all 4 gestures combined. We also observed two marked differences in the accuracy of the classifier between the smartphone data and Glass data. First, the accuracy of the system on all the swipe gestures on the smartphone is better than Glass. However, this might be due to the fact that the total number of swipe gestures were smaller, i.e., 20, in the smartphone data. Secondly, the accuracy of the system is less impacted with increasing number of users on smartphone than Glass. A plausible reason for these two differences might be due to the difference in touchpad size of the two devices. Bigger touchpad size allows for more variation in the gesture patterns. It is interesting to investigate whether other gesture-based authentication mechanisms proposed for smartphones exhibit a similar trend on smart glasses.

5.7 Effect of Behavioural Evolution on Classification Accuracy

As the gesture behaviour of users may change over time, we studied its evolution through an extended study on three users asking them to use Glass for five days over two weeks. The five days were spaced as: day 1, 2, 3, 7 and 14. We used a fixed training size of 20. To test the permanence of a user's gesture model, We experimented with the following three scenarios related to how the training model was generated. (a) *Same Day*: This scenario serves as the benchmark.

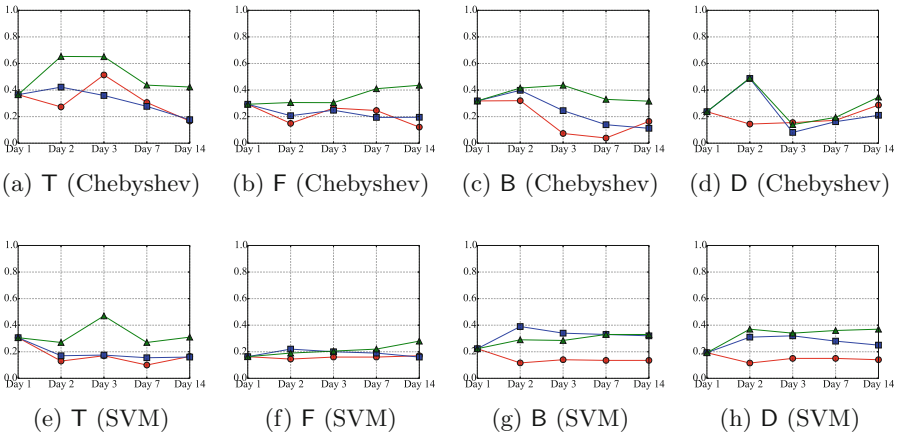
The testing data is matched against training data collected from the same day. (b) *First Day*: In this scenario, each user model is generated using data from day one. The model is then tested against data collected on subsequent days. For instance, day seven against day one. (c) *Adaptive*: In this scenario, the user model is updated every day, by iteratively replacing a fixed number of samples in the training data of previous days with random samples from the data of the same day. For example, to create the training data for day 3, we randomly replaced 8 samples from the training data of day one with 4 samples from day two and 4 samples from day three.

For the Chebyshev classifier, for each simulation run we use one random user as the target user and the remaining two as the attack users. In case of the SVM classifier, each of the three user is taken as a target user. The training data for the target user consists of a random sample of a fixed size from the target user's data. This constitutes positive samples for the target user required for SVM training. The negative samples for the target user come from the data of the remaining two target users. The attackers' data come from a fixed set of three users who did not participate in the evolution study and whose data was collected for earlier experiments. The results are shown in Fig. 6 for both the classifiers. As expected, the same day scenario achieves the highest accuracy amongst all the scenarios for a given day. We can also observe that the accuracy of the first day scenario is the worst, suggesting that the touch biometrics are not quite stable over time and hence an adaptive approach should be considered to maintain accuracy over time. Using adaptive approach in our experiments clearly shows performance improvements over the first day scenario, especially for the Chebyshev classifier. Note that replacing older samples with newer ones means that the classifiers need to be re-trained. For the Chebyshev classifier, this is not an issue since re-training takes around 1 s at worst (cf. Table 7). For SVM, training takes longer, but this is not a substantial hurdle due to the reasons discussed in Sect. 5.5.

6 Some Limitations and Discussion

We did not consider the effect of user posture, e.g., walking versus sitting, on touch gestures. Although this difference may not be as profound as in the case of smartphones, since the Glass is mounted on the user's head and is relatively stable, it needs to be experimentally determined. Since the focus of our research has been touch gesture based continuous authentication, we have overlooked voice characteristics (as mentioned before, the user can also perform certain operations in Glass through voice commands) or readings from other sensors such as accelerometer and gyroscope. Our continuous authentication system can be augmented by including distinguishing features from voice or other sensors. Also, as is the case for any behaviour biometric system, it is important to test our system on the larger population to validate its accuracy, a feat we were unable to perform due to limited resources.

Since the Chebyshev classifier is based on a concentration inequality, it will be interesting to employ other concentration inequalities such as Hoeffding or



(a) T (Chebyshev) (b) F (Chebyshev) (c) B (Chebyshev) (d) D (Chebyshev)

(e) T (SVM) (f) F (SVM) (g) B (SVM) (h) D (SVM)

Fig. 6. The evolution of EER - Chebyshev classifier and AER - SVM classifier. Legend: ● *same day* training data; ■ *adaptive* training data; ▲ *first day* training data.

Bernstein's inequalities to compare the results. As a classifier's performance is also dependent on the features being used, it will be interesting to expand on the feature model introduced in this paper. For instance, one may model the swipe feature as an interaction between the two forces (downward and planar), instead of taking the two forces separately. A resulting feature could be a three dimensional magnitude of force over time.

7 Conclusion

Due to smaller touchpad size and relatively meagre resources of current smart glasses hardware (CPU, battery) compared to modern smartphones, it is not straightforward to assume that gesture based implicit authentication systems proposed for smartphones would yield high classification accuracy and low computational load on smart glasses, such as Google Glass. The results of our study indicate that gesture based continuous authentication is indeed both computationally and accuracy-wise feasible on Glass. Among other contributions of our work is the proposal of a new classifier based on Chebyshev's concentration inequality, which can be added to other classifiers used in the field of implicit authentication. Our secondary contributions include modelling touch gestures in a new way from which we extract new features such as downward (as measured by pressure and area readings) and planar (as measured by velocity readings) force as a function of time, and the finding that classification accuracy is dependent on the size of the touchpad.

References

1. Amidan, B.G., Ferryman, T.A., Cooley, S.K.: Data outlier detection using the chebyshev theorem. In: IEEE Aerospace Conference, pp. 3814–3819 (2005)

2. Ben-Asher, N., Kirschnick, N., Sieger, H., Meyer, J., Ben-Oved, A., Möller, S.: On the need for different security methods on mobile phones. In: MobileHCI 2011, pp. 465–473. ACM (2011)
3. Bo, C., Zhang, L., Li, X.Y., Huang, Q., Wang, Y.: SilentSense: silent user identification via touch and movement behavioral biometrics. In: MobiCom 2013, pp. 187–190 (2013)
4. Burgbacher, U., Hinrichs, K.: An implicit author verification system for text messages based on gesture typing biometrics. In: CHI 2014, pp. 2951–2954. ACM (2014)
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27:1–27:27 (2011)
6. De Luca, A., Hang, A., Brudy, F., Lindner, C., Hussmann, H.: Touch me once and i know it's you!: implicit authentication based on touch screen patterns. In: CHI 2012, pp. 987–996. ACM (2012)
7. Frank, M., Biedert, R., Ma, E., Martinovic, I., Song, D.: Touchalytics: on the applicability of touchscreen input as a behavioral biometric for continuous authentication. IEEE Trans. Inf. forensics Secur. **8**(1), 136–148 (2013)
8. Giuffrida, C., Majdanik, K., Conti, M., Bos, H.: I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics. In: Dietrich, S. (ed.) DIMVA 2014. LNCS, vol. 8550, pp. 92–111. Springer, Heidelberg (2014)
9. Jakobsson, M., Shi, E., Golle, P., Chow, R.: Implicit authentication for mobile devices. In: HotSec 2009, p. 9. USENIX (2009)
10. Li, L., Zhao, X., Xue, G.: Unobservable re-authentication for smartphones. In: NDSS 2013. Internet Society (2013)
11. LiKamWa, R., Wang, Z., Carroll, A., Lin, F.X., Zhong, L.: Draining our glass: an energy and heat characterization of google glass. In: APSys 2014, pp. 10:1–10:7. ACM (2014)
12. Mondal, S., Bours, P.: Swipe gesture based continuous authentication for mobile devices. In: ICB 2015, pp. 458–465. IEEE (2015)
13. Ross, S.M.: A First Course in Probability, 4th edn. Prentice Hall, Upper Saddle River (2002)
14. Schaub, F., Deyhle, R., Weber, M.: Password entry usability and shoulder surfing susceptibility on different smartphone platforms. In: MUM 2012, pp. 13:1–13:10. ACM (2012)
15. Serwadda, A., Phoha, V.V., Wang, Z.: Which verifiers work?: a benchmark evaluation of touch-based authentication algorithms. In: BTAS 2013, pp. 1–8. IEEE (2013)
16. Shahzad, M., Liu, A.X., Samuel, A.: Secure unlocking of mobile touch screen devices by simple gestures: you can see it but you can not do it. In: MobiCom 2013, pp. 39–50. ACM (2013)
17. Stanciu, V.D., Spolaor, R., Conti, M., Giuffrida, C.: On the effectiveness of sensor-enhanced keystroke dynamics against statistical attacks. In: CODASPY 2016, pp. 105–112. ACM (2016)
18. Uellenbeck, S., Dürmuth, M., Wolf, C., Holz, T.: Quantifying the security of graphical passwords: the case of android unlock patterns. In: CCS 2013, pp. 161–172. ACM (2013)
19. Xu, H., Zhou, Y., Lyu, M.R.: Towards continuous and passive authentication via touch biometrics: an experimental study on smartphones. In: SOUPS 2014, pp. 187–198. ACM (2014)
20. Zheng, N., Bai, K., Huang, H., Wang, H.: You are how you touch: User verification on smartphones via tapping behaviors. In: ICNP 2014, pp. 221–232. ACM (2014)