Rahat Masood*, Benjamin Zi Hao Zhao, Hassan Jameel Asghar, and Mohamed Ali Kaafar

Touch and You're Trapp(ck)ed: Quantifying the Uniqueness of Touch Gestures for Tracking

Abstract: We argue that touch-based gestures on touch-screen devices enable the threat of a form of persistent and ubiquitous tracking which we call *touch-based tracking*. Touch-based tracking goes beyond the tracking of virtual identities and has the potential for cross-device tracking as well as identifying multiple users using the same device. We demonstrate the likelihood of touch-based tracking by focusing on touch gestures widely used to interact with touch devices such as swipes and taps.. Our objective is to quantify and measure the information carried by touch-based gestures which may lead to tracking users. For this purpose, we develop an information theoretic method that measures the amount of information about users leaked by gestures when modelled as feature vectors. Our methodology allows us to evaluate the information leaked by individual features of gestures, samples of gestures, as well as samples of combinations of gestures. Through our purpose-built app, called TouchTrack, we gather gesture samples from 89 users, and demonstrate that touch gestures contain sufficient information to uniquely identify and track users. Our results show that writing samples (on a touch pad) can reveal 73.7% of information (when measured in bits), and left swipes can reveal up to 68.6% of information. Combining different combinations of gestures results in higher uniqueness, with the combination of keystrokes, swipes and writing revealing up to 98.5% of information about users. We further show that, through our methodology, we can correctly re-identify returning users with a success rate of more than 90%.

Keywords: Implicit Tracking, Mobile Privacy, Touch Gestures, User Profiling, Risk Quantification

DOI Editor to enter DOI
Received ..; revised ..; accepted ...

***Corresponding Author: Rahat Masood:** CSIRO Data61 and University of New South Wales (UNSW), Sydney, Australia, E-mail: rahat.masood@student.unsw.edu.au
Benjamin Zi Hao Zhao: CSIRO Data61, Sydney, Australia, E-mail: ben.zhao@data61.csiro.au
Hassan Jameel Asghar: CSIRO Data61, Sydney, Australia, E-mail: hassan.asghar@data61.csiro.au

1 Introduction

Touch gestures such as swipes, taps and keystrokes, are common modes of interaction with smart touchscreen-enabled devices, e.g., smartphones, smart watches and smart glasses. Major platforms including Android OS, iOS, watchOS and Android Wear provide a variety of APIs to help app developers detect gestures aiming to enhance the quality of experience of apps. Access to these APIs allows apps to collect raw gesture data from different sensors available on the smart device. The fine-grained nature of this data means that there is potential of learning more about users than is perhaps necessary for the proper functioning of an app. Indeed one area where touch gestures have been exploited is continuous authentication through which users are authenticated by profiling their touch behaviour [1, 14].

In this paper, we argue and verify that touch gestures constitute a privacy threat as they enable a new form of tracking of individuals, which we refer to as "touch-based tracking," which is the ability to continuously and surreptitiously track and distinguish users via their touch behaviour while they are interacting with their devices. As compared to "regular" tracking mechanisms, e.g., based on cookies, browser fingerprints, browser user agents, logins and IP addresses, several factors make touch-based tracking potentially riskier. First, while regular tracking tracks virtual identities such as online profiles [13, 20, 25], touch-based tracking has the potential to track and identify the actual (physical) person operating the device. It can distinguish and track multiple users accessing the same device. Second, touch-based tracking possesses the capability to *continuously* track users. Third, it also leads to cross-device tracking where the same user can potentially be traced on multiple mobile devices. Cross-device tracking introduces additional privacy and security risks, where user data can be collated and sent to advertising companies and third party entities to build user profiles based on

Mohamed Ali Kaafar: Macquarie University, Optus Macquarie University Cyber Security Hub, and CSIRO Data61, Sydney, Australia, E-mail: dali.kaafar@mq.edu.au

their activities on smartphones, tablets, smart watches and various IoT devices. However, demonstrating this type of tracking requires a more generalized approach, e.g. to validate the stability of features across devices, which we leave as future work.

Not all use cases of touch-based tracking are negative. It can also be beneficial to users and service providers alike. For instance, the identification of multiple users using the same device may help in providing content more suitable for each of them. A child using his/her parent's smartphone can automatically have parental control enabled. Touch-based tracking could also bring commercial benefits to the user (e.g. displaying discounts and sales on the product of interest to the user). The reader might notice a link between touch-based tracking and touch-based continuous authentication. There are major differences in the two notions. The latter verifies a claimed identity based on prior knowledge of the identity and former tracks users even without the knowledge of any disclosed identity. We elaborate on this and other differences between the two in Section 5.

The ubiquity of the touchscreen devices and the fact that most if not all data from touch events and/or other sensors can be extracted by any mobile app without requesting special permission makes touch-based tracking a serious privacy threat for users. This not only represents a valuable new source of information for analytics, tracking, and ad services but also for app developers who can (mis)use the information to track individuals on a single device or across multiple devices. The objective of this paper is to quantify the amount of information carried by user's touch gestures and hence to evaluate their tracking capabilities. Our main contributions are summarised as follows.

- We investigate the potential of using touch-based gestures for tracking, which we call touch-based tracking. We quantify the amount of information contained in these gestures which could lead to user tracking. To the best of our knowledge, this is the first study considering the potential of touch gestures to profile users. Our work complements research on other forms of tracking such as through web browsers, host devices, and online social profiles by fingerprinting browser features, device configurations, and user attributes [4, 7, 13, 18, 19, 24, 25, 27, 38].
- We develop an **analytical framework** that measures the amount of identifying information (in bits and relative mutual information) contained in touch gestures, represented as feature vectors, at different levels of granularity. At the finest level, our framework quantifies the information carried by individual features, e.g., pressure on screen and area covered by the gesture. At the second level, our framework quantifies the information carried by a gesture sample, e.g., a single swipe. At the third level, our framework calculates the amount of information carried by multiple samples of the same gesture, e.g., a collection of swipes. Lastly, we measure the information carried by a collection of samples from multiple gestures, e.g., swipes and taps. We apply our framework on four widely used touch screen gestures: i) swipes, ii) taps, iii) keystrokes, and iv) handwriting, and four sub-categories of swipe: i) up swipe, ii) down swipe, iii) left swipe, and iv) right swipe. The framework is generic enough to apply to any behavioural biometric modality which can be expressed as feature vectors.
- We **develop and deploy a game-like app** called "TouchTrack" for Android powered devices. It consists of three well known open source games: 2048 (for swipes),[1] Lexica (for taps),[2] Logo Maniac (for keystrokes),[3] and one custom built app for handwriting. These games were selected to capture touch gestures in a natural way. Through our TouchTrack app the user can check the uniqueness and tracking potential of his/her gestures.
- Using our TouchTrack app, we carried out a user study comprised of 89 participants and gathered a total of 40,600 samples of touch gestures. For each touch gesture, we **identified features that contain high amount of identifying information** using the *maximum-relevancy minimum-redundancy* (mRMR) algorithm [26]. The algorithm attempts to constrain features to a subset which are mutually dissimilar to each other, but similar to the classification variable, which in our case was the set of users. We give details in Section 4.2. We found that the most revealing features were the 80th percentile of area from left swipes, the 20th percentile of area and the 50th percentile of pressure from downward swipes which yielded 56.1%, 55.50% and 46.13% of information, respectively.
- With the same dataset, we **measured the amount of information contained in samples from the**

1 https://github.com/gabrielecirulli/2048
2 https://github.com/lexica/lexica
3 https://github.com/Luze26/LogoManiac

same gesture and from multiple gestures.[4] We found that 50 features in a single handwriting sample contribute 68.71% of information about users, which increases to 73.7% with multiple samples. We further identified that two or three different gestures combined together reveal more information about users. For instance swipes, handwriting, and keystrokes carry 98.5%, while handwriting, taps, and swipes disclose 95.1% of information. Among users who performed all the four gestures, our framework revealed 98.89% of information about users.

– Finally, we also **validated our framework in terms of correctly identifying a returning user**. This is important since the same user might have two different samples from the same gesture that could be mutually dissimilar (thus showing high uniqueness) but will not result in identifying the user. We measure the true positive and false positive rates (TPR and FPR) of our method. We define TPR as the rate at which a unique user (in our set of users) is identified as the *correct* user given a test sample (or set of samples). Likewise, FPR is defined as the rate at which the wrong user is identified as the target user or a set of more than one users is identified as the set of possible users given a test sample (or set of samples). We found that with multiple samples, swipes and handwriting show a TPR of 90.0% and 91.0%, respectively. For a combination of gestures we found that swipes and handwriting combined together had a TPR of 93.75%. In terms of FPR, we found that swipes, handwriting, and keystrokes taken together had an FPR of only 0.8%.

Overall our results demonstrate that touch gestures can be used to uniquely identify (or fingerprint) users with high accuracy. This illustrates the threat of tracking based on touch gestures in general. The rest of the paper is organized as follows: Section 2 covers our methodology of collecting data using TouchTrack app, and then presents the descriptive statistics about our dataset. Section 3 outlines our proposed probabilistic analytical framework in detail. In Section 4, we discuss the results on the amount of information conveyed by the users of our dataset for different touch gestures and their combinations . We discuss the limitation and future directions of this work in Section 5. Finally, we describe related work in Section 6 and conclude in Section 7.

4 We use the relative mutual information as our metric for identifying information. For details, see Section 3.

2 Data Collection

To illustrate the potential of touch-based tracking, we developed and launched a purpose-built app named *TouchTrack* to capture gesture samples. We first give an overview of the TouchTrack app, followed by our data collection approach. We then briefly describe some statistics about our dataset.

2.1 Selection of Gestures

Our selection of gestures was based on how frequently they are performed by users of smartphones or other touchscreen devices. We narrowed our selection to swipes (including left, right, upward and downwards swipes, and the group of four taken together), taps, keystrokes and handwriting. Swipes and taps are by far the most frequent gestures on smartphone apps. Keystrokes are also frequently used for typing text messages or entering web addresses. Unlike tap, which could be performed at any point on the touch screen, a keystroke is restricted to tapping on the mobile keyboard. We therefore separated the two. Writing on the touchscreen using fingers is an important alternative input method on a smartphone. We did not include other less frequent gestures such as pinching (for zooming in or out).

2.2 The TouchTrack App

The purpose of TouchTrack is to collect gesture samples as raw readings from the touch sensors, send them to our server, and finally inform the users about the uniqueness of their gestures by displaying the results computed via our framework at the server. To keep the user interested in using our app, we decided to design it like a game. The app is made up of four games, three of them are based on popular open-source games and a fourth game was purposely developed by us. We selected these four games so as to capture the user gestures in a most natural way. We briefly describe each game in Appendix A.1 along with the screenshots of the games.

When a new user uses TouchTrack, he/she is required to sign up using a unique username. The username together with the device ID is hashed and then stored in our database. This is done to ensure that gesture samples from different users are kept separate to establish the ground truth. This also helps us to identify returning users and devices. For matters of ease and privacy, we did not require the user to enter a password with the username. Even though our app sends data in HTTPs, we did not want to collect any additional sensitive data, such as passwords, from users. Once the user has played one or more games, the uniqueness of the corresponding gestures are computed through our quantitative framework (described in Section 3) and are shown to the user in both visual and tabular forms. Screen shots of results are shown in Figure 7 of Appendix A.1. For user convenience, our app starts showing results after a single gesture. However, to get more accurate results, the user is encouraged to perform more gestures. We would like to remark that our results may still not be reflective of user touch behaviour in the real world, as displaying uniqueness results might encourage the user to change touch behaviour to avoid privacy leakage. Probable change in user behaviour due to feedback has been acknowledged before in the case of browser-based tracking [13]. We have made the TouchTrack app available on Google Play Store as our objective is to extend our work in the future to assess the effectiveness of touch-based tracking as more users are added.

2.3 The Raw Dataset

To collect gesture samples, we invited participants through two means: via emails to targeted participants and via social networking platforms. At first we uploaded a closed testing version of TouchTrack on Google Play Store and invited colleagues and acquaintances via email to install our app. A total of 25 participants responded to the first phase. In the second phase, we published the first version on Google Play Store and received a further 56 responses through our personal and social contacts. We received 8 responses from the users who installed our app without invitation. We also included them in our analysis. The app was available on Google Play Store for two months for data collection purposes. Once the data is collected, we start our analysis and results interpretation. The data collected from the 89 users served two purposes: to identify features most effective in fingerprinting users and to train our analytical framework to evaluate the uniqueness of gestures.

Table 1 shows the list of raw touch features gathered from the touch sensors of the devices used by the users across all gestures. By default, these features can be obtained from Android APIs without requiring any security permission. We used the `MotionEvent` Android API to detect and collect touch data. We did not use motion features for our analysis because we observed that certain motion sensors such as accelerometer, gyroscope and rotation vector did not produce any raw data in many phones, and returned a null value to the Android API.

2.4 Ethics Consideration

Prior to data collection, we underwent and obtained an ethics approval from our organization's ethics board. The users were informed about the purpose of TouchTrack and what data is being collected. Throughout data collection, we did not attempt to obtain the real identities of the participants via, for instance, a linkage study. The data collected was not released publicly. No identifying information other than the user selected username and device ID was stored at our server side. Moreover, only the hash of the username and device ID were stored. The only other information stored at the server were the raw data from gestures from each username-device ID pair. A privacy disclaimer was displayed as soon as the app was launched by a participant providing the above details. The participant was allowed to opt-out. We informed users that their personal information will remain anonymous and took their consent beforehand.

2.5 Data Statistics

Table 2 shows the summary statistics of our data collection. The numbers are broken down into number of users, samples of gestures and features associated with each gesture. We collected a total of 40,600 gesture samples. Among these samples, swipe and tap gestures had the most number of samples. There were a total of 89 users who downloaded and used our app; however, only 30 users used all four games and hence provided samples for all gestures. Our app was installed on 49 different smart phone models, with *Google Nexus 5* being used by 11 users and *Samsung Galaxy S7 Edge* by 8 users. Nine of the 11 users of Google Nexus 5 used our test smartphone to record gesture samples as they did not have an Android powered device. We could distinguish

Table 1. Raw Features

Raw Touch Features	X-Coordinate, Y-Coordinate, Finger Pressure, Finger Area, Screen Orientation, Finger Orientation, Stroke Time, X-Tilt , Y-Tilt

Table 2. Touch Gesture Data Statistics

Gesture	Number of Users	Number of Samples	Number of Features
Swipes	81	16611	229
Up Swipes	78	3568	229
Down Swipes	71	4781	229
Left Swipes	63	4252	229
Right Swipes	65	4010	229
Taps	89	16225	7
Keystrokes	49	6473	8
Handwriting	36	1291	241
All Gestures:	30	25186	
Total:	89	40600	

between users of the same device via their hashed user ID.

3 Methodology for Computing the Uniqueness of User Gestures

In this section we describe our methodology behind computing uniqueness of gestures. We begin with an overview, followed by notations and then the methodology in detail.

3.1 Overview

Recall that the purpose of calculating uniqueness is to demonstrate touch-based tracking. For this, we need to show (a) the uniqueness of gestures, (b) similarity of gestures from the same user. To do this, we first obtain gesture samples, i.e., series of raw data values captured from the sensors of the smart device from a set of users. Once these samples are collected we extract a set of salient features, thus representing each gesture by a feature vector. The set of selected features is the topic of Section 4.1. For now we assume that each gesture is associated with a fixed set of features. Once we have populated our dataset with an initial list of users and gesture samples as instances of feature vectors, we then proceed to find the uniqueness of the gestures at different levels. At the smallest level, we assess the uniqueness of single features, by checking how many users exhibit a given 'test' feature value among the total users in the dataset. At the next level we assess the uniqueness of a feature vector, i.e., a gesture sample, by checking how many users in the dataset are likely to exhibit a given test feature vector. Likewise, we do this for a collection of samples from the same gesture, and finally for the collection of samples from a set of different gestures. In what follows, we define an abstract representation of our dataset, and how we compute the uniqueness of gestures at multiple levels using this abstract dataset.

3.2 Background and Notations

We denote the set of users by $\mathbb{U}$, the set of gestures by $\mathbb{G}$, and the feature space by $\mathbb{F}$. We denote our dataset by D which is modelled as a multiset of rows. The columns of D are indexed by a $u \in \mathbb{U}$, followed by a $g \in \mathbb{G}$, a feature vector $\mathbf{f} \in \mathbb{F}$, and finally by an average feature vector $\overline{\mathbf{f}} \in \overline{\mathbb{F}}$. The average feature vector $\overline{\mathbf{f}}$ is the average of all feature vectors $\mathbf{f}$ under a gesture $g \in \mathbb{G}$ and a user $u \in \mathbb{U}$. The ith feature under $\mathbb{F}$ is denoted by $\mathbb{F}_i$. The dataset is illustrated in Table 3. We define a random variable U that takes on values from $\mathbb{U}$, a random variable G that takes on values of subsets of $\mathbb{G}$, and a random variable F_g that takes on values of subsets of feature vectors from the gesture g. When considering only a single gesture g, we shall drop the subscript and denote the random variable as F. We use the notation $\{a\}$ to indicate a set of cardinality more than 1, whose generic element is denoted by a. For instance, $\{\mathbf{f}\}$ is a set of two or more feature vectors. The random variable F can take feature values as well. Abusing notation, we will denote this by $F = f_i$, where f_i is the value of the ith feature. A predicate on a row from D denoted

$$(\mathbb{U} = u, \mathbb{G} = g, \mathbb{F}_1 = f_1, \mathbb{F}_2 = f_2, \ldots, \overline{\mathbb{F}} = \overline{\mathbf{f}}),$$

is the conjunction of clauses $(\mathbb{U} = u)$, $(\mathbb{G} = g)$, and so on. The predicate evaluates to 1 if a row satisfies each clause, and 0 otherwise. We can have possibly empty clauses. When considering a feature vector, we may simply use $\mathbb{F}$ to represent the conjunction of its constituent clauses. For example, the predicate

$$(\mathbb{U} = \text{Alice}, \mathbb{G} = \text{Swipe}, \mathbb{F} = (0.012, 0.567, *, *, \ldots, *)),$$

evaluates to 1 on the first row of Table 3, where '$*$' indicates that the corresponding feature values are not part of the predicate. A fuzzy predicate is a function that evaluates to 1, if the feature vector of a row is similar to the feature vector specified in the clauses according to a similarity metric. Fuzzy predicates are distinguished from predicates by replacing either the equality involving $\mathbb{F}$ or $\overline{\mathbb{F}}$ by $\approx$. For instance, the following is a fuzzy predicate

$$(\mathbb{U} = \text{Alice}, \mathbb{G} = \text{Swipe}, \mathbb{F} \approx (0.012, 0.567, \ldots, 0.314)).$$

We denote by $\#(\cdot)$ the number of rows in D satisfying the predicate (or fuzzy predicate). The entropy of the random variable U is defined as

$$\begin{aligned} H(U) &= -\sum_{u \in \mathbb{U}} \Pr(U = u) \log_2 \Pr(U = u) \\ &= -\sum_{u \in \mathbb{U}} \frac{1}{|\mathbb{U}|} \log_2 \frac{1}{|\mathbb{U}|} = \log_2 |\mathbb{U}|. \end{aligned}$$

This is the minimum number of bits of information required to distinguish each user in $\mathbb{U}$. The mutual information or information gain between U and a realization a of the random variable A is defined as

$$I(U; A = a) = H(U) - H(U \mid A = a),$$

where $H(U \mid A = a)$ is the conditional entropy given as

$$\begin{aligned} H(U \mid A = a) = -\sum_{u \in \mathbb{U}} &\Pr(U = u \mid A = a) \\ &\times \log_2 \Pr(U = u \mid A = a). \end{aligned} \tag{1}$$

Finally, the relative mutual information between a realization a of the random variable A is defined as

$$I_R(U; A = a) = \frac{I(U; A = a)}{H(U)} = 1 - \frac{H(U \mid A = a)}{H(U)}. \tag{2}$$

Table 3. Structure of the Dataset D.

$\mathbb{U}$	$\mathbb{G}$	$\mathbb{F}$			$\overline{\mathbb{F}}$
		$\mathbb{F}_1$	$\mathbb{F}_2$	$\cdots$	
Alice	Swipe	0.012	0.567	$\cdots$	$(0.021, 0.770, \cdots)$
		0.019	0.599	$\cdots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
Bob	Tap	0.023	0.608	$\cdots$	$(0.010, 0.660, \cdots)$
		0.024	0.499	$\cdots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	

The above measures the uniqueness of a realization of a random variable A through relative mutual information. To assess the uniqueness of all possible values the random variable A can take, we make use of the conditional entropy

$$H(U \mid A) = -\sum_{a \in A} \Pr(A = a) H(U \mid A = a) \tag{3}$$

Here, $\Pr(A = a)$ is calculated from the probability distribution of the random variable A. From this, the relative mutual information of the random variable A is

$$I_R(U; A) = 1 - \frac{H(U \mid A)}{H(U)}. \tag{4}$$

Note that while mutual information should suffice as a measure to assess uniqueness, our choice of relative mutual information is to account for the different number of users for different gestures, thus enabling us to compare results across gestures on the same scale. In what follows, we shall assess the uniqueness based on different realizations of the random variables G and F. This will be done by first calculating the conditional probabilities in Eq. 1 which are determined by predicates or fuzzy predicates, then computing the conditional entropy in Eq. 1, which then directly allows us to compute the relative mutual information in Eq. 2. A given realization is considered highly unique if the relative mutual information is close to 1. We will use percentages to represent the value of relative mutual information in the range $[0, 1]$ in the natural way. To assess the uniqueness of the random variables G and F in its entirety, we will make use of the relative mutual information defined by Eq. 4.

3.3 Measuring Uniqueness

We measure uniqueness based on a single feature value from a gesture sample, a single feature vector (i.e., a gesture sample), a set of feature vectors (i.e., a set of gesture samples), and finally a set of feature vectors corresponding to a set of gesture samples from multiple gestures. To measure uniqueness based on a single continuous feature, we first bin its values within discrete bins and then calculate the probability of a user producing the feature value within a bin. In contrast, to evaluate uniqueness of features vector(s), we do not bin the features, and instead rely on fuzzy predicates. Our procedure to bin continuous features (for evaluating uniqueness of single features) is described in Appendix A.2.

3.3.1 Uniqueness based on a Feature value

Given a feature value f_i corresponding to the ith feature of a gesture $g \in \mathbb{G}$, the uniqueness of the value is computed as follows. We first calculate the probability that a $u \in \mathbb{U}$ is likely to have produced this feature value. This probability is calculated by Eq 5.

$$\Pr(U = u \mid G = g, F = f_i) = \frac{\#(\mathbb{U} = u, \mathbb{G} = g, \mathbb{F}_i = f_i)}{\#(\mathbb{G} = g, \mathbb{F}_i = f_i)} \qquad (5)$$

The conditional entropy in U given the feature value f_i of a sample of the gesture g is given by plugging the above conditional probability in Eq. 1 to obtain $H(U \mid G = g, F = f_i)$, from which the relative mutual information $I(U; G = g, F = f_i)$ can be obtained from Eq. 2.

Example 1. Suppose our dataset has $|\mathbb{U}| = 128$ users, giving us $H(U) = \log_2 |\mathbb{U}| = 7$ bits. Suppose now we are looking at the swipe gesture, and we are interested in the first feature having value $f_1 = 0.012$. Further suppose that out of the 128 users, only Alice and Bob have exhibited this value in the dataset, with Alice having exhibited it twice (corresponding to two different samples of swipe), and Bob only once. We have $\#(\mathbb{G} = \text{Swipe}, \mathbb{F}_1 = 0.012) = 3$, $\#(\mathbb{U} = \text{Alice}, \mathbb{G} = \text{Swipe}, \mathbb{F}_1 = 0.012) = 2$, and $\#(\mathbb{U} = \text{Bob}, \mathbb{G} = \text{Swipe}, \mathbb{F}_1 = 0.012) = 1$. Then $\Pr(U = \text{Alice} \mid G = \text{Swipe}, F = 0.012) = \frac{2}{3}$ and $\Pr(U = \text{Bob} \mid G = \text{Swipe}, F = 0.012) = \frac{1}{3}$. From this we get $H(U \mid G = \text{Swipe}, F = 0.012) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} \approx 0.92$ And finally, $I_R(U; G = \text{Swipe}, F = 0.012) = 0.8688$. We say that the feature value $f_1 = 0.012$ for the swipe gesture reveals 87% of information. □

To assess the uniqueness of the ith feature (and not just one particular feature value) we calculate the probability that the random variable F corresponding to the ith feature takes on the feature value f_i as

$$\Pr(F = f_i \mid G = g) = \frac{\#(\mathbb{G} = g, \mathbb{F}_i = f_i)}{\sum_{f \in F} \#(\mathbb{G} = g, \mathbb{F}_i = f)}. \qquad (6)$$

That is we count all instances of the feature value f_i and divide it by the sum of all instances of feature values f in the range of F. By plugging this value and the result of conditional entropy of feature values in Eq. 3, we obtain the conditional entropy pertaining to F, from which we can compute the relative mutual information $I(U; F)$ from Eq. 4.

3.3.2 Uniqueness based on a Gesture Sample

To measure uniqueness of a gesture sample, we use the entire feature vector $\mathbf{f}$ corresponding to the gesture g, and check against all feature vectors of the user u. Due to high dimensionality of the feature vector, it is unlikely that any two feature vectors from the same user will be exactly the same. We therefore use the fuzzy predicate in this case, which relies on a similarity metric. We postpone our choice of the similarity metric to Section 3.4. The conditional probability is calculated as

$$\Pr(U = u \mid G = g, F = \mathbf{f}) = \frac{\#(\mathbb{U} = u, \mathbb{G} = g, \mathbb{F} \approx \mathbf{f})}{\#(\mathbb{G} = g, \mathbb{F} \approx \mathbf{f})}, \qquad (7)$$

From this probability we can then compute the conditional entropy and relative mutual information as before. Due to space limit we omit how the relative mutual information for the entire gesture is computed, which is similar to the case of the relative mutual information for a feature.

3.3.3 Uniqueness based on a Set of Gesture Samples

If we are given a set of feature vectors $\{\mathbf{f}\}$ from a gesture g, we first obtain the average vector $\overline{\mathbf{f}}$ from $\{\mathbf{f}\}$. Then, we compare this average vector against the average vector under $\overline{F}$ of the user $u \in U$ (for the same gesture). Given this, the probability that the set of gesture samples is from the user $u \in U$ is

$$\Pr(U = u \mid G = g, F = \{\mathbf{f}\}) = \frac{\#(\mathbb{U} = u, \mathbb{G} = g, \overline{\mathbb{F}} \approx \overline{\mathbf{f}})}{\#(\mathbb{G} = g, \overline{\mathbb{F}} \approx \overline{\mathbf{f}})} \qquad (8)$$

Notice the use of fuzzy predicates. Given this probability, the conditional entropy and relative mutual information can be computed as before.

3.3.4 Uniqueness based on Multiple Gestures

Given a subset of gestures $\{g\}$ and their corresponding sets of feature vectors $\{\mathbf{f}_g\}$, we first obtain an average feature vector for each gesture, denoted $\overline{\mathbf{f}}_g$, and then count the number of rows in D that satisfy the product of the fuzzy predicates generated by the average feature vectors of the gestures involved. More specifically, the probability of the collection belonging to a user $u \in \mathbb{U}$ is calculated as

$$\Pr(U = u \mid G = \{g\}, \{F_g = \{\mathbf{f}_g\}\}) = \frac{\prod_g (\mathbb{U} = u, \mathbb{G} = g, \overline{\mathbb{F}} \approx \overline{\mathbf{f}}_g)}{\sum_{u' \in U} \left(\prod_g (\mathbb{U} = u', \mathbb{G} = g, \overline{\mathbb{F}} \approx \overline{\mathbf{f}}_g)\right)} \qquad (9)$$

The symbol $\prod$ stands for product. In this case the product is over all gestures in $\{g\}$. For instance, if we have $\{\text{Swipe}, \text{Tap}\}$ as two gestures, then we are essentially checking if the product predicate

$$(\mathbb{G} = \text{Swipe}, \overline{\mathbb{F}} \approx \overline{\mathbf{f}}_{\text{Swipe}}) \times (\mathbb{G} = \text{Tap}, \overline{\mathbb{F}} \approx \overline{\mathbf{f}}_{\text{Tap}})$$

evaluates to 1, which is only possible if both the fuzzy predicates evaluate to 1 for a given user in D. We divide this by summing the same product predicate for all users in D. The conditional entropy and relative mutual information can be computed by plugging in the above conditional probability.

3.4 Calculating Fuzzy Predicates

To demonstrate how fuzzy predicates are evaluated, we use a generic feature vector $\mathbf{f}$ belonging to a gesture $g \in \mathbb{G}$. This is tested against a feature vector $\mathbf{f}'$ belonging to a row in D under $\mathbf{F}$ or $\overline{\mathbf{F}}$ (in case of the latter, we have an average feature vector). As mentioned before, evaluation of the fuzzy predicate is tied to a similarity metric. We chose the cosine similarity metric. Assume the length of $\mathbf{f}$ is m, then the cosine of the angle between $\mathbf{f}$ and $\mathbf{f}'$ is defined as

$$\cos(\mathbf{f}, \mathbf{f}') = \frac{\sum_{i=1}^{m} f_i f_i'}{\sqrt{\sum_{i=1}^{m} f_i^2}\sqrt{\sum_{i=1}^{m} f_i'^2}}, \tag{10}$$

which ranges between −1 and 1, the latter indicating complete similarity. Together with a threshold $\tau \in [-1, 1]$, the cosine similarity metric is then

$$s_{\cos}(\mathbf{f}, \mathbf{f}', \tau) = \begin{cases} 1, & \text{if} \cos(\mathbf{f}, \mathbf{f}') \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

Example 2. Given a row $(u', g', \mathbf{f}', \cdot)$ of the dataset D (ignoring the last column under $\overline{\mathbf{F}}$), the fuzzy predicate $(\mathbb{U} = u, \mathbb{G} = g, \mathbb{F} \approx \mathbf{f})$ evaluates to 1 if $u' = u$, $g' = g$ and $s_{\cos}(\mathbf{f}, \mathbf{f}', \tau) = 1$. □

The threshold τ is set by balancing the true positive rate (TPR) and the false positive rate (FPR), i.e., the value that returns the best equal error rate (EER). Details on this appear in Section 4.5.

4 Results

In this section, we present and discuss the results of applying our framework to show the uniqueness of gestures. Our goal is to show that touch gestures can be used to track users. For this, we need to show (a) that they are highly unique and (b) their ability to identify returning users. We first identify a set of features for each gesture.Then we rank the features in terms of their distinguishing capacity and finally we apply our methodology on the selected features to show uniqueness results.

4.1 Feature Identification and Extraction

From the raw features described in Table 1 (cf. Section 2.3), we derived more features to capture information such as averages, standard deviations, minimums and maximums. These derived features are called extracted features. A gesture sample generates a sequence of raw data points. The length of this sequence depends on the duration of the gesture and the *sampling rate* (usually around a millisecond), which is normally different across devices. This means that the sequences corresponding to two samples from the same gesture may not be of the same length. We therefore performed spline polynomial interpolation [10] to ensure the same number of data points (length of sequence) across samples from the same gesture. Since the sequences from different gestures are expected to be of different lengths, we did a separate interpolation for each gesture.

We identified a set of most commonly used features in literature (on gesture based authentication). We extracted 229 features for swipes, 7 for taps, 8 for keystrokes, and 241 for handwriting. Out of these, only 7 features are common across all gesture categories. These features are Inter-Stroke Time, Stroke Duration, Start X, Start Pressure, Start Y, Start Area, and Mid-Stroke Finger Orientation. Table 9 in Appendix A.5 shows the list of these features. A few of the extracted features are Median of First 5 Acceleration Points, 80-percentile of pairwise X-Tilt, Std. Dev. of Pairwise Change of Area-Position, Direct End to End Direction, End to End X Distance, Median of Last 3 Velocities Points, 20-percentile pairwise Pressure etc.

4.2 Feature Subset Selection (FSS)

As a first step, we were interested in finding the uniqueness of gestures as a function of increasing number of features. To do this, we needed a ranking of features in terms of their distinguishing capacity.We use the maximum-relevance-minimal-redundancy (mRMR) algorithm that attempts to constrain features to a subset which are mutually as dissimilar to each other as possible, but as similar to the classification variable as possible [26]. In our case, the classification variable is the set $\mathbb{U}$ of users in our dataset. Given a set of m features $F_1, \ldots, F_m$ to find the highest rank feature, the mRMR algorithm finds the feature F_i that maximizes $I(U; F_i)$, where U takes on values from $\mathbb{U}$, and minimizes $I(F_i; F_j)$, where $j \neq i$. The mutual information $I(F_i; F_j)$ is calculated by using the joint distribution of features F_i and F_j through our dataset. Likewise, for more than one feature, the algorithm returns the subset of features that maximize, respectively minimize, the cumulative mutual information. A more detailed description of the algorithm is given in Appendix A.3.

4.3 Effect of Number of Features on Uniqueness

In order to determine uniqueness of gestures as a function of features, we used sets of top i features from each gesture according to their mRMR rank, where i was incremented in discrete steps until m (the total number of features). We then evaluated their relative mutual information using our framework for the uniqueness of a single gesture sample (cf. Section 3.3.2) and multiple samples from the same gesture (cf. Section 3.3.3).

To do this we first partitioned the data from each user-gesture pair into two random but mutually exclusive sets. The first set had 80% of the gesture samples, and the second had the remaining 20% of the samples. The larger set was labelled as our dataset D, and samples from the 20% set were used for "testing." We shall call this approach the 80-20 approach throughout the rest of this paper. We call the 20% set, the testing set. Thus, to evaluate our methodology, we select a sample from the testing set (fixing a user and a gesture), and then check against the dataset D.

Now to check the effect of an increasing number of features on the uniqueness of gestures, we selected top i features from the mRMR ranking for incremental values of i and then used the above mentioned 80-20 partitioning. We used an exhaustive approach, i.e., for testing single samples, we selected each sample in the the testing sets of all users, and then calculated the relative mutual information using Eq. 4. For testing a set of gesture samples, we used the entire 20% testing set as one set of gesture samples, and subsequently computed the relative mutual information. In our methodology, the relative mutual information for both these categories requires evaluating the fuzzy predicate, which in turn is determined by the threshold τ of the cosine similarity metric. For these results we set a universal threshold of $\tau = 0.8$, since we wanted to check the effect of mutual information keeping everything else fixed. The outcome of this analysis is depicted in Table 4. We note that for all gestures, the relative mutual information increases with increasing number of features. Also, the uniqueness of a set of gesture samples is generally higher than single samples, and in all cases surpasses the uniqueness of single samples as we increase the number of features. The uniqueness of multiple swipe samples is the highest, with 92.01% (highlighted green with *), followed by handwriting (85.93%) and downward swipes (77.52%). On the other hand, samples of taps and keystrokes exhibit least uniqueness carrying 34.73% and 41.02% of information. This may also be due to the low number of features identified for these gestures. We observe that given a single gesture sample, handwriting provides 79.49% (highlighted green with *) of information about the user and a keystroke gives the least amount of information i.e. 28.76%.

The above analysis does not take into account the true positive rate (TPR) of the uniqueness measurement. Given a test sample from a user $u \in \mathbb{U}$, we mark it as a true positive if the corresponding predicate only evaluates to 1 on the user u. Otherwise, we mark it as a false positive. Note that this means that if the predicate evaluates to 1 for more than one user, then we consider it as a false positive even if it evaluated to 1 on the user u. The TPR and FPR are then evaluated over all possible test samples. Table 5 shows the TPRs and FPRs for different sets of features corresponding to a single sample and multiple samples from a gesture. We found that TPR decreases with the increase in number of features. The most probable reason for this continual decrease is the variations in a user's own touch behaviour as the dimension of the feature space increases.

With only 15 features, the TPR is 89% or above for all gestures (with multiple samples). In terms of FPR rates, we see that keystrokes and taps have relatively high FPR rates (43% and 37%, respectively, for multiple samples). The FPR decreases as we increase the number of features. We selected first 50 features for handwriting, 50 for swipes (all four types), 8 for keystrokes, and 7 for taps (highlighted blue in Tables 4 & 5) for further analysis, as these presented a good balance between TPR and FPR. We also performed 10-fold cross-validation and splits such as 30-70, 40-60 using Weka to evaluate the relative mutual information, TPR and FPR of a single gesture sample and multiple samples from the same gesture. The results gathered from these

Table 4. Relative Mutual Information for a varying set of features. Green cells with * indicate highest relative mutual information for a gesture sample and a set of gesture samples. Blue highlighted rows indicate our final selection of features.

Gesture	# of	Rel. Mutual Information of		Gesture	# of	Rel. Mutual Information of	
	Features	Gesture Sample	Set of Gesture Samples		Features	Gesture Sample	Set of Gesture Samples
Swipe	15	43.23%	40.09%	Left Swipe	15	47.71%	52.89%
(Expected	20	45.53%	41.91%	(Expected	20	50.59%	55.59%
IS: 6.32 bits)	25	46.09%	45.40%	IS: 5.97 bits)	25	50.71%	56.85%
	30	48.18%	45.60%		30	52.38%	60.03%
	50	57.79%	63.39%		50	53.96%	68.66%
	75	61.39%	75.34%		75	57.96%	70.62%
	100	61.87%	83.28%		100	59.82%	71.11%
	150	62.88%	88.50%		150	62.64%	71.55%
	200	63.13%	91.23%		200	65.52%	74.23%
	229	64.10%	92.01%*		229	65.77%	74.68%
Up Swipe	15	45.93%	43.30%	Right Swipe	15	48.29%	53.71%
(Expected	20	48.05%	46.39%	(Expected	20	50.03%	54.14%
IS: 6.28 bits)	25	48.26%	46.59%	IS:6.02 bits)	25	50.44%	56.00%
	30	48.56%	47.43%		30	51.24%	56.19%
	50	49.02%	50.23%		50	52.27%	57.48%
	75	55.09%	63.81%		75	55.59%	62.62%
	100	58.68%	68.79%		100	56.58%	65.35%
	150	58.74%	69.22%		150	57.11%	65.88%
	200	61.53%	71.94%		200	59.88%	67.48%
	229	61.68%	73.11%		229	60.12%	67.65%
Down Swipe	15	48.46%	46.85%	Handwriting	15	47.06%	52.24%
(Expected	20	51.44%	49.89%	(Expected	20	49.35%	52.78%
IS: 6.14 bits)	25	51.53%	51.17%	IS:5.16 bits)	25	52.93%	55.94%
	30	51.60%	51.43%		30	55.57%	58.99%
	50	52.22%	54.58%		50	68.71%	73.72%
	75	58.33%	67.51%		75	72.09%	77.19%
	100	60.55%	70.05%		100	74.75%	79.34%
	150	62.33%	71.35%		150	78.08%	83.47%
	200	65.25%	75.59%		200	78.16%	85.36%
	229	65.51%	77.52%		241	79.49%*	85.93%
Keystroke	1	23.92%	17.83%	Tap	1	24.80%	15.17%
(Expected	2	26.29%	20.00%	Expected	2	26.25%	25.23%
IS:5.61 bits)	3	26.62%	29.97%	IS:6.47 bits)	3	27.85%	26.94%
	4	26.86%	30.49%		4	28.75%	33.25%
	5	26.86%	30.49%		5	29.48%	34.25%
	6	27.25%	36.70%		6	29.55%	34.44%
	7	27.34%	37.63%		7	29.58%	34.73%
	8	28.76%	41.02%				

approaches were similar to 20-80 approach. We are thus mentioning results from 20-80 approach only.

4.4 Uniqueness of Individual Features

Before assessing the uniqueness of features we binned any continuous features or features with a large domain. See Appendix A.2 for details. To assess the uniqueness of features, we again divided our dataset using the aforementioned 80-20 partition. Then, we exhaustively computed the relative mutual information defined in Eq. 4 as a measure of uniqueness for each feature value in the testing sets of all users. We found that *80-percentile of area* in left swipe reveals 56.10% of information about a user, followed by *20-percentile of area* in down swipe 55.50%. Similarly, *50-percentile of pressure* yields 46.13% of information from down swipe. Among features which are shared among all gestures,*start area* contains 52.5% of information, followed by *start pressure* yielding 45.4% of information. On the other extreme, *inter-stroke time* for a keystroke reveals minimum amount of user information, i.e,. 7%. We observe no trend (in terms of dependency) among features,except that relative information decreases in decending order of the features. We also computed the cumulative distribution function (CDF) of the relative mutual information through Eq. 2 for a given feature. We present the CDF of top five features of every gesture in figure 1. It is evident that features corresponding

Table 5. TPR and FPR of Gesture for a varying number of features. Green cells with * indicate highest TPR and low FPR for a gesture sample and a set of gesture samples. Blue highlighted rows indicates our final selection of features.

Gesture	# of	Gesture Sample		Set of Gesture Samples		Gesture	# of	Gesture Sample		Set of Gesture Samples	
	Features	TPR	FPR	TPR	FPR		Features	TPR	FPR	TPR	FPR
Swipe	15	0.67	0.15	0.92	0.20	Left Swipe	15	0.55	0.07	0.91	0.14
	20	0.67	0.14	0.94	0.20		20	0.53	0.07	0.90	0.12
	25	0.62	0.11	0.94	0.17		25	0.51	0.06	0.86	0.12
	30	0.57	0.11	0.92	0.16		30	0.46	0.04	0.86	0.11
	50	0.23	0.02	0.89	0.07		50	0.34	0.02	0.83	0.07
	75	0.14	0.009	0.89	0.03		75	0.33	0.02	0.81	0.07
	100	0.11	0.007	0.89	0.02		100	0.31	0.02	0.77	0.07
	150	0.10	0.007	0.84	0.01		150	0.30	0.02	0.78	0.07
	200	0.10	0.007	0.76	0.009		200	0.29	0.02	0.77	0.06
	229	0.10	0.006*	0.76	0.009*		229	0.29	0.02	0.75	0.06
Up Swipe	15	0.56	0.10	0.89	0.16	Right Swipe	15	0.55	0.08	0.93	0.13
	20	0.54	0.10	0.89	0.14		20	0.54	0.06	0.88	0.12
	25	0.52	0.09	0.88	0.14		25	0.53	0.06	0.86	0.12
	30	0.52	0.09	0.88	0.14		30	0.51	0.06	0.86	0.11
	50	0.51	0.08	0.85	0.13		50	0.51	0.05	0.85	0.10
	75	0.40	0.05	0.82	0.07		75	0.41	0.03	0.85	0.09
	100	0.37	0.05	0.79	0.05		100	0.41	0.03	0.85	0.08
	150	0.37	0.04	0.79	0.05		150	0.41	0.03	0.85	0.08
	200	0.35	0.04	0.77	0.05		200	0.39	0.03	0.83	0.08
	229	0.34	0.04	0.74	0.04		229	0.39*	0.03	0.83	0.08
Down Swipe	15	0.57	0.11	0.92	0.17	Handwriting	15	0.67	0.11	0.97	0.16
	20	0.53	0.09	0.90	0.14		20	0.64	0.10	0.94	0.16
	25	0.53	0.08	0.88	0.14		25	0.47	0.05	0.92	0.14
	30	0.53	0.08	0.87	0.14		30	0.47	0.04	0.89	0.13
	50	0.49	0.08	0.85	0.12		50	0.29	0.01	0.88	0.06
	75	0.40	0.04	0.85	0.07		75	0.24	0.01	0.82	0.05
	100	0.38	0.04	0.84	0.06		100	0.21	0.009	0.79	0.05
	150	0.34	0.03	0.84	0.05		150	0.15	0.006*	0.73	0.04
	200	0.32	0.03	0.82	0.04		200	0.14	0.006*	0.64	0.03
	229	0.32	0.03	0.77	0.04		241	0.13	0.006*	0.61	0.03
Keystroke	1	0.46	0.23	0.96	0.43	Tap	1	0.54	0.26	0.95	0.37
	2	0.44	0.21	0.96	0.40		2	0.53	0.22	0.94	0.35
	3	0.43	0.18	0.95	0.37		3	0.53	0.21	0.94	0.34
	4	0.41	0.17	0.93	0.36		4	0.32	0.10	0.85	0.27
	5	0.40	0.17	0.93	0.36		5	0.31	0.10	0.85	0.27
	6	0.22	0.09	0.83	0.32		6	0.31	0.10	0.85	0.26
	7	0.22	0.09	0.83	0.32		7	0.31	0.10	0.85	0.26
	8	0.18	0.08	0.79	0.27						

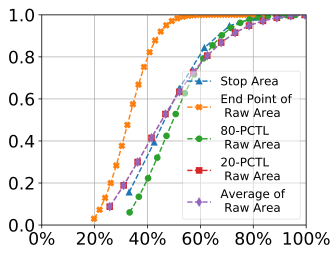
to different statistics of *area* and *pressure*, e.g., average, percentiles etc., reveal the most information about a user, i.e., more than 40% of information for half of the users in our database. As before, we notice that all types of swipes and handwriting reveal the most information about users, and taps and keystrokes have relatively less information leakage about users.

Table 6. Thresholds of the cosine similarity metric for a gesture sample. τ = Threshold, EER = Equal Error Rate.

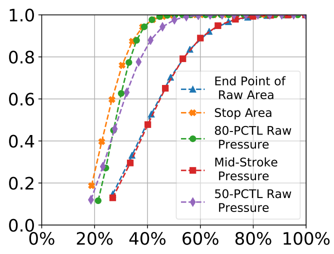
Gesture	τ	EER	Gesture	τ	EER
Swipe	0.38	22%	Up Swipe	0.55	27%
Down Swipe	0.52	28%	Left Swipe	0.48	22%
Right Swipe	0.58	22%	Handwriting	0.40	19%
Tap	0.29	35%	Keystroke	0.13	39%

4.5 Uniqueness of a Gesture Sample

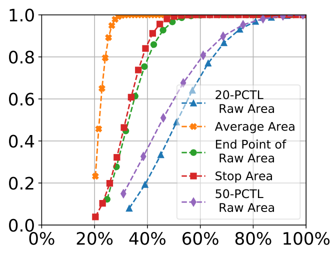
Recall from Section 3.3.2 that for gesture sample, we need to calculate the fuzzy predicate using the cosine similarity metric (cf. Section 3.4). Once we have fixed the set of features, and hence fixed the feature space, we need to find the threshold τ of the cosine similarity metric that balances uniqueness of gesture samples and correctly (and uniquely) identifying a returning user. To



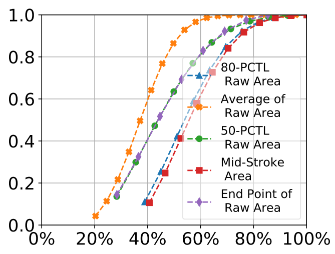
(a) Swipes



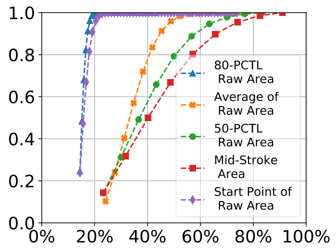
(b) Up Swipe



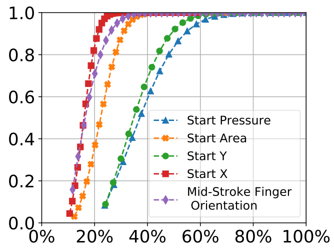
(c) Down Swipe



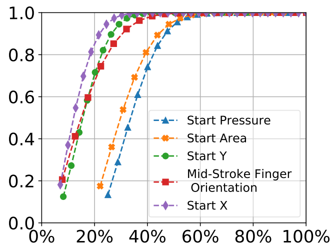
(d) Left Swipe



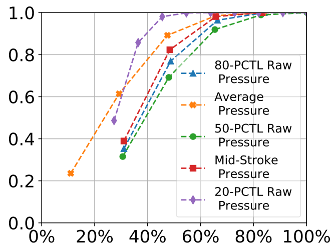
(e) Right Swipe



(f) Tap



(g) Keystroke



(h) Handwriting

Fig. 1. Cumulative Distribution Function (CDF) of Features. Y-axes represents fraction of the participant population and X-axes are Relative Mutual Information in percentage. The graph shows that Swipe, Left and Down Swipe reveals more than 50% of information for half of the population, respectively.

do this, we once again split the data into an 80-20 partition, and then evaluated the equal error rate (EER) (i.e., the rate at which $1 - \text{TPR}$ equals FPR) by varying the threshold. Table 6 shows the threshold that gave the lowest EER against each gesture. We can see that our methodology correctly re-identifies a returning user up to 81% (19% EER) of the time if given a handwriting sample. The worst performance is a TPR of 61% (39% EER) when a sample of keystroke is provided. The ROC of a gesture sample for all gesture types is given in Apendix A.5.

After fixing the threshold, we computed the uniqueness through our relative mutual information metric, i.e., Eq. 3. The results showed that a handwriting sample reveals the highest amount of information (68.71%), followed by swipes (57.77%). The four types of swipes, i.e., left, up, down, and right swipes, yield 53.9%, 52.2%, 52.2%, and 48.5% of information, respectively. However, taps and keystroke reveal only 29.5% and 26.2% of information, respectively. Figure 2 shows the CDF of a gesture sample calculated for each of the gesture (through the relative mutual information metric of Eq. 2). We observe a considerable difference in the range of information revealed by different gestures, with handwriting exposing more than 60% of information for half of the users in the database. Following this, the swipes also show high uniqueness, revealing 30% to 65% of information about 75% of users. This suggests that handwriting and swipes carry highly identifiable information about users.

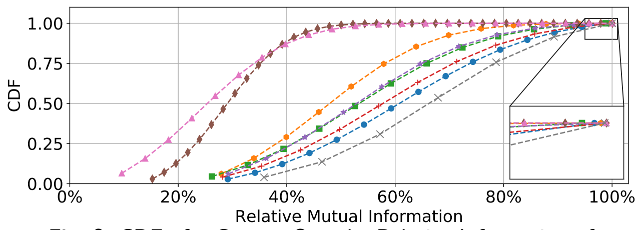


Fig. 2. CDF of a Gesture Sample. Relative Information of Respective Categories are: -●- Swipe: 57.7%, -⬡- Up Swipe: 48.5%, -■- Down Swipe: 52.2%, -+- Left Swipe: 53.9%, -★- Right Swipe: 53.3%, -◆- Tap: 29.5%, -▲- Keystroke: 26.2%, -×- Handwriting: 68.7%

4.6 Uniqueness of a Set of Gesture Samples

We now consider the amount of information revealed by multiple samples of each gesture. We computed a different threshold of the cosine similarity metric for this category, and chose the one which resulted in the best EER. Table 7 shows the threshold and the corresponding EER values. Comparing this table to Table 6, we see that the rate of re-identifying a returning user is higher reaching up to 91% (9% EER) for handwriting. This means that combining a few samples of the same gesture may allow for more accurate tracking. The ROC of a set of gesture samples for all gesture types are given in Appendix A.5.

Table 7. Thresholds of the cosine similarity metric for a set of gesture samples. τ = Threshold, EER = Equal Error Rate.

Gesture	τ	EER	Gesture	τ	EER
Swipe	0.75	10%	Up Swipe	0.77	16%
Down Swipe	0.78	14%	Left Swipe	0.75	12%
Right Swipe	0.77	12%	Handwriting	0.76	09%
Tap	0.85	20%	Keystroke	0.85	23%

Based on the threshold values obtained, we then apply the cosine similarity metric on the dataset and calculate relative mutual information through Eq. 4. Once again handwriting reveals 73.7% of information, followed by left swipe which yields 68.6% of information of user gestures. In accordance with previous results, taps and keystrokes reveal minimum amount of information about users, i.e., 34.71% and 41.0%, respectively. Looking at the CDF of relative mutual information in Figure 3, we can observe that swipes, its subtypes, and handwriting consistently perform better in revealing information than taps and keystrokes. As discussed earlier, the less information from keystrokes and taps can be due to the less number of features identified for these gestures.

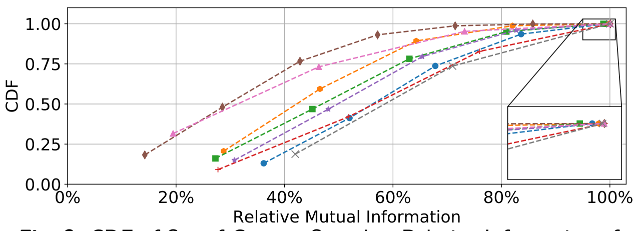


Fig. 3. CDF of Set of Gesture Samples. Relative Information of Respective Categories are: -●- Swipes: 63.3%, -⬡- Up Swipes: 50.23%, -■- Down Swipes: 54.5%, -+- Left Swipes: 68.6%, -★- Right Swipes: 57.4%, -◆- Taps: 34.7%, -▲- Keystrokes: 41.0%, -×- Handwriting: 73.7%

4.7 Uniqueness of Gesture Categories Combination

Next we consider multiple gestures in different combinations and measure their uniqueness through our methodology outlined in Section 3.3.4. Figure 4 shows the quantification results for different gesture combinations. We found that a combination of all gestures reveal a maximum of 98.89% of information about users, followed by the combination of swipes, handwriting & keystrokes that yield 98.5% of information. In contrast, the combination of taps and keystroke reveals minimum information, i.e., 33.5%. We would like to emphasise here that information revealed by the combination of various gestures is dependent on the number of users who had performed all gestures in the combination. This number was different for different gesture combinations. For example, the total number of users who performed taps was 89, whereas only 49 users submitted keystroke samples (cf. Table 2). Furthermore, the total number of users who had performed both taps and keystrokes were 45. This is one reason for our choice of the relative mutual information metric (as opposed to simple mutual information) which "normalises" the mutual information.

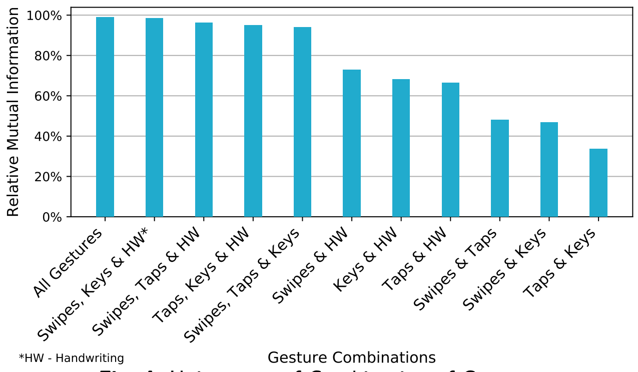


Fig. 4. Uniqueness of Combination of Gestures

We also tested these gesture combinations in terms of re-identifying returning users. The thresholds for the cosine similarity metric for each gesture were as reported in the previous section (Table 7). Figure 5 shows the TPR and FPR of the different combinations of gestures. Since the threshold for the cosine similarity metric for each gesture was already set, the figure does not report EER as TPR and FPR are not balanced. For this reason, we also show the true negative and false negative rates. We see that as we increase the number of gestures in our combination, the FPR drastically decreases, but so does the TPR. For instance, all gestures together yield 0.99% FPR but also a low TPR (just above 40%). The lowest FPR was recorded by the combination of swipes, handwriting and keystrokes (0.85%). The main reason for a big drop in TPR as compared to the rate of single gestures, is mainly due to the rather strict metric of only labelling a given combination as being from a user if the predicate for each gesture evaluates to 1 (cf. Sec-

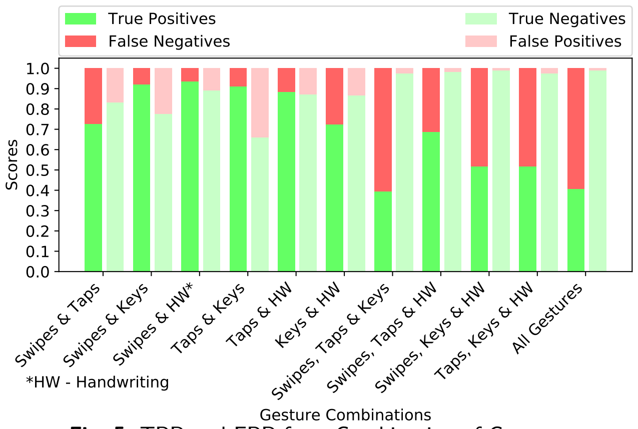


Fig. 5. TPR and FPR for a Combination of Gestures

tion 3.3.3). This can be changed by using, for instance, a majority rule.

We also investigate the impact of different users using the same device however, due to space limitation we present our results in appendix A.4.

5 Discussion

Our results reveal some important findings which we enlist below.

1. Multiple samples of a gesture taken together reveal more accurate information about a user than a single sample of a gesture. This means that tracking based on collecting larger number of user samples is likely to be more accurate. Having said that a single gesture sample or a single feature of a gesture also reveal enough information (upto 68% and 56%, respectively) so as significantly narrow down the set of possible users for tracking.
2. Swipes and handwriting carry more information as compared to taps and keystrokes. This is largely due to the rich set of information that can be derived as features from swipes and handwriting. In contrast, taps and keystrokes are simpler gestures from which only a few characteristic features can be derived.
3. Features based on the area and pressure of the finger performing the gesture are the most informative. This shows that there is significant inter-user variation in the area covered and the pressured exerted on the screen.
4. Overall, we show that tracking using touch gestures is highly feasible and accurate. This is demonstrated by the fact that we can correctly identify a returning user with a true positive rate of up to 91% with the handwriting samples. Though, TPR is a critical metric for an authentication system that evaluates the system's ability to correctly identify users. However, in case of tracking, a near 100% TPR is less critical, e.g., it may be acceptable for an advertising company to correctly track 90% of users and display related ads, whilst showing unrelated ads to 10%. This is indeed the case with our scheme, where 9% of users will be incorrectly classified (receive irrelevant ads). Still, our methodology would correctly classify 91% of the users to display relevant ads. In future, we plan to expand the framework to improve the classification results, and hence minimise the FPR.
5. Our data collection procedure did not impose any condition on how users needed to interact with their smartphones such as sitting, standing, and walking postures. Our results are still able to show high uniqueness and accurate re-identification of returning users.

Touch-based Tracking vs. Continuous/Implicit Authentication:

The reader might confuse the notion of touch-based tracking with touch-based continuous or implicit authentication. Even though the main goal of this paper is to quantify the amount of information carried by touch gestures and hence to evaluate the tracking capabilities using touch based features, in the following we would like to clarify some major differences between the two notions. Conceptually, the goal of authentication is to verify a claimed identity which assumes prior knowledge of the identity. The aim of touch-based tracking is to track users with or without the knowledge of any disclosed identity. Here we highlight further technical differences.

1. A typical continuous authentication scheme involves a classifier which is trained on the data of a target user.[5] This training data is gathered during a preceding registration phase. The classifier knows which training model to target for classification given a claimed identity. Touch-based tracking on the other hand is supposed to function without knowing the identity of the current user. This implies no registration phase and therefore the absence of training models for target users. Thus, classifica-

5 Or a set of users using the same device, in which case each user has a separate training model. See for instance [37].

tion methods used for continuous authentication are not readily applicable for touch-based tracking.

2. Continuous authentication schemes require a certain number of samples before an authentication decision can be made with confidence. This is less of a stringent requirement on touch-based tracking, and tracking may proceed with even a single observation. The probable user set may be large, but it does not hinder tracking. Therefore, classification methods used in continuous authentication schemes are too restricted for use in touch-based tracking.
3. As a corollary to above, high classification rate, i.e., near 100% TPR and low FPR, is critical for the success of a continuous authentication scheme. In the case of tracking, a high TPR or misclassification rate is less critical, e.g., it may be acceptable for an advertising company to correctly track 90% of users and display related ads, whilst showing unrelated ads to 10%.
4. A final point is around measuring uniqueness. The goal of touch-based tracking is to illustrate how tracking is probable. This involves measuring how touch gestures, for instance, convey unique information. This needs to be measured at all levels: from single features to a collection of samples from multiple gestures. Our goal is to demonstrate how different granularity of information contained in gestures contribute to uniqueness and subsequent tracking of users. Some of this information does not lead to (successful) continuous authentication, e.g., uniqueness of single features. On the other hand, for touch-based tracking, this information is useful as it can be used in conjunction with other information (not necessarily from gestures) to more accurately track users.

In light of the above, we argue that touch-based tracking requires a different methodology from continuous authentication systems (to assess uniqueness of touch-based gestures).

Limitations:
We have only used the cosine similarity metric to evaluate uniqueness. We have not investigated other similarity metrics such as Euclidean distance, manhattan distance, and Jaccard index for comparison. Our main goal was to demonstrate the feasibility of touch-based tracking, and for that fixing a representative similarity metric was sufficient. Our quantitative methodology can also be extended by replacing the similarity metric with a machine learning classifier such as support vector machines (SVM) or k-nearest neighbours (kNN).

Our focus in this paper has been on four commonly used touch gestures. It is possible that other not-so widely used gestures such as pinch-in, pinch-out, drag, touch and hold, and multi-finger touches may lead to better unique identification rates of users and consequently user tracking. Another important aspect for future investigation is to verify our methodology for more complex scenarios such as *single-device multi-user tracking* and *multi-device single-user tracking*. The first scenario distinguishes between multiple users accessing the same device, e.g., a smartphone in a family accessed by parents and kids. The second scenario is the tracking of the same user across multiple devices. This scenario is quite realistic as the average number of connected devices per person is 3.5 [3]. Therefore, tracking user across multiple devices can create a potential risk to user privacy. Cross-device tracking is not straightforward to evaluate as it requires a more generalized approach. For example, it requires selecting "stable" features across all types of devices. This requires significantly more work in data collection and measurement to validate the stability of features across devices, which we intend to work on in the future.

The stability of the gesture based fingerprint with time also needs to be investigated, as the user behaviour normally changes with time and it may have an impact on the accuracy of the uniqueness. We also need to verify the reliability and accuracy of our methodology as more users and more gesture data is collected through out TouchTrack app. Moreover, we did not at present take motion sensor features into account. There is likely a possibility that user uniqueness increases with the addition of these features. Our framework heavily relies on raw features extracted from android API; it assumes that raw features can be accessed from API's without requiring security permissions. While this assumption is valid for now, a number of other ways could be identified and used to extract raw features from mobile API's. Finally, we did not apply our framework on other mobile operating systems (OS) such as iOS and Windows, as we cannot access the raw features from these OS without having security permissions.

6 Related Work

Several techniques, in the past, have been proposed on implicit/continuous authentication. Frank et al.

[14] presented an Implicit Authentication (IA) scheme named 'Touchalytics' to authenticate user based on their finger movements on a touch screen of a smartphone. Their scheme used mutual information to select features that give high information gain. Sherman et al. [32] studied free-form multitouch gestures for mobile authentication. A part of their work used mutual information to illustrate the security strength of free-form gestures. However, they did not apply mutual information directly to gesture data. Instead, they first preprocess the data to remove any predictable information from the gestures (using second order autoregressive model). The residual of each gesture sample thus obtained, is used to measure the mutual information between two gesture samples. The overall purpose was to assess the security strength of free-form gestures for which they use mutual information to weed out any predictable information content in those gestures. Contrary to this, we directly apply mutual information on the gesture samples as our goal is to measure the amount of information carried by touch gestures, and to see how uniquely identifiable a user is based on the information contained in gestures. Secondly, mutual information is not further used for authentication in the work by Sherman et al. In contrast, in our work, we use the mutual information metric (relative mutual information to be precise), to measure information at all levels including combinations of gesture samples.

Like wise, there are a number of other touch gesture based continuous/implicit authentication schemes proposed in [1, 23, 28, 31, 37, 40, 41]. Keystroke dynamics for authenticating mobile phone users have also been widely studied [15, 17, 21, 36, 39]. A body of work has also been conducted on Unobtrusive methods for IA on mobile phones such as Gait-based IA schemes [11, 35] and context aware schemes [5, 16, 33]. Our work differs significantly from the above mentioned schemes based on the reasons given in Section 5.

Device and browser fingerprinting has been numerously demonstrated by the research community. Perhaps the pioneering work in the threat of tracking dates back to Sweeney, who showed for the first time that coarse-grained information such as birthday, gender, and ZIP code can uniquely identify a person [34]. This work was followed by several studies that provided measurement insights into web and device tracking. Eckersley [13] quantified the uniqueness of web browsers based on user agent and/or the browser configuration (plugins, fonts, cookies, screen resolution etc.). Olejnik et al. [25] performed a large-scale analysis of web browsing histories to track web users. Similarly, a number of device and browser tracking methods have been demonstrated in [20, 22, 38].

A body of work also focused on fingerprinting mobile device. Studies by Kurtz [19] and Bojinov [2] focus on different physical characteristics of a mobile device, with the former focusing on device configurations and the latter utilizing the noisy nature of hardware sensors such as accelerometer and microphones. Similarly, work performed in [6–9, 12, 42] focused on fingerprinting devices based on different physical characteristics. A number of studies has also focused on identifying mobile user traits and characteristics using the information provided by mobile SDKs to third party apps, such as list of installed apps, running apps, device model, operating systems etc. [19, 30].

Our work differs from the abovementioned techniques which are confined to fingerprinting virtual identities. For-instance, the browser fingerprinting techniques utilized characteristics such as plug-ins, fonts, caches, histories etc. to uniquely identify a browser. Likewise, keystroke dynamics-based tracking focuses either on keyboard-equipped devices or keystroke based gestures only. By comparison, our demonstration of touch-based tracking can track the physical identity of a person by fingerprinting the touch gestures on a mobile device used by the same person or the same device used by different users.

7 Conclusion

In this paper, we introduce a new privacy threat induced by the collection and monitoring of touch gestures of mobile device users. We proposed and developed an analytical framework that quantifies the amount of information carried by the user touch gestures mainly swipes, keystrokes, taps, and handwriting. We quantify uniqueness at four different levels from a feature value to the combinations of gestures altogether. In addition, we also developed an android app, called TouchTrack, that collects users gesture data and provides real-time results about the uniqueness of their gestures.

Our findings highlight that user touch gestures exhibit high uniqueness. Additionally, we also showed that returning users could be correctly re-identified with high accuracy, indicating that touch-based tracking is possible. We plan to extend our framework to demonstrate other aspects of touch-based tracking such as cross-device user tracking.

A Appendix

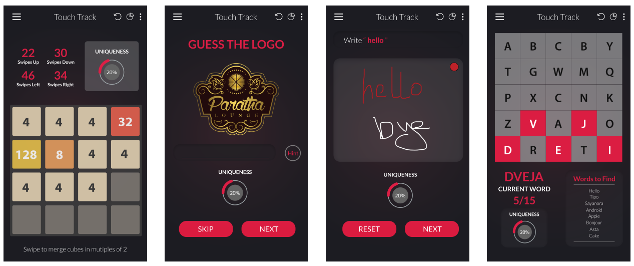
This supplementary section gives more insights on the information presented in above sections.

A.1 TouchTrack App Overview

The TouchTrack app consists of three well-known games and one purpose-built game. A brief description of games are given below:

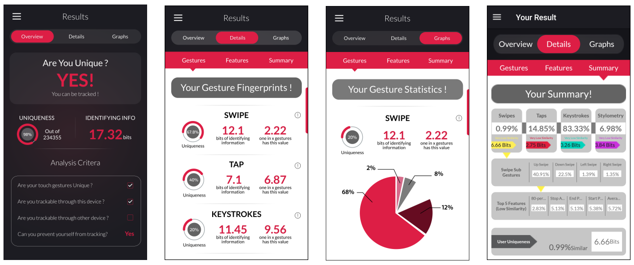
1. *2048:* We used this game to collect swipes. It is a free and open-source game which is played on a 4 by 4 grid having numbered tiles that need to be swiped in any of the four directions. We selected this game since it is widely known and it captures swipes mimicking their usage in a natural way, i.e., while reading emails or swiping through an image gallery.
2. *Lexica:* We used this game to collect taps. It is another open-source free word game that gives the user three minutes to find as many words as possible on a 5 by 5 grid of random letters. The original behaviour of the game requires user to drag letters to make a single word. For our work, we changed the drag operation to a tap, and ask user to tap on the letter to select it, or tap on again the same letter for de-selection. The grid of 5*5 allows user to tap on almost every point of screen, thus simulating natural taps.
3. *Logo Maniac:* We used this game to collect keystrokes. The game tests the user's ability to recall popular brands by showing logos and asking them to type the brand name. We modify this game by only having the most popular brands in our database, and providing hints to user if they cannot recall it. We modified the keyboard layout of the game to make it similar to the keyboard layout used for entering texts in Android phones, to capture the user's natural typing behaviour on phones.
4. *Write Something:* We used this game to collect handwriting samples. This game was purpose-built by us. It asks users to write a word shown at top left corner of the screen with a finger. User is provided with a large area on the screen to write in any direction or from any point.

The screen shots of the TouchTrack App are displayed in figure 6, while figure 7 shows the shots of result screen. We show uniqueness results for a feature, set of gesture samples, and multiple gestures to our app users.



(a) 2048 (b) Logo Maniac (c) Write Something (d) Lexica

Fig. 6. TouchTrack Game Screens



(a) Results Overview (b) Gesture Results (c) Graphs (d) Summary

Fig. 7. TouchTrack Result Screens

A.2 Binning Feature Values

If a feature is continuous, then the probability that its corresponding random variable F has the value f is 0. In an actual implementation, continuous features are replaced by their floating point analogues. Still, the probability that the random variable exhibits the exact value f is negligibly small. This will result in our uniqueness based measure returning every feature value as unique (even from the same user). We therefore, bin features that are either continuous or have a large domain. Let σ denote standard deviation. Fix a gesture $g \in G$, let $n = \#(G = g)$ denote the number of samples of the feature. We use Scott's formula [29] to obtain the optimum bin size Δf as

$$\Delta f = \frac{3.49\sigma(F)}{n}.$$

Given this bin width the total number of bins are then

$$\left\lceil \frac{\max F - \min F}{\Delta f} \right\rceil.$$

Given a feature value f, its bin is calculated as

$$b = \left\lceil \frac{f - \min F}{\Delta f} \right\rceil$$

The feature value f is then converted to the feature value

$$\hat{f} = b\Delta f + \min F$$

The value $\hat{f}$ is then stored in the dataset D instead of f.

A.3 The mRMR Algorithm and Results

We intend to select features which have a potential to uniquely identify users based on touch gestures. In order to select the most distinguishing and non-redundant features from the given list, mRMR defines the subset of features that maximizes mutual information with the class label $I(F;C)$ and minimizes the information between $I(f_i;f_j)$[6] To apply mRMR, one must convert the features to discrete variables. We discretize features using bin's approach and use scott's rule to get equally spaced bins. We input a list of features

```
Input: Set of all features $S_{\text{tot}} = \{f_1, f_2, ......f_k\}$, a
       gesture $\mathbf{g} \in \mathbb{G}$, and a Class $\mathbf{c}$.
1  Initialize $F_{\text{sel}} \leftarrow \varnothing$, $\beta \leftarrow \frac{1.0}{|S_{\text{tot}}|}$.
2  for $i = 1$ to $|S_{\text{tot}}|$ do
3      Compute feature relevancy $I(f_i;c)$ using
       $I(S,c) = I(\{f_i, i = 1, 2, ...k\}; c)$
4      Select feature which has the highest value i.e.
       $\max(I(S;c))$
5  if $F_{sel} == \varnothing$ then
6      Append $F_{\text{sel}} \leftarrow F_{\text{sel}} + f_{max}$
7      Set $f_{sel} \leftarrow \max(I(S;c))$
8  if $F_{sel} == |S_{\text{tot}}|$ then
9      break
10 for $j = 1$ to $|S_{\text{tot}}|$ do
11     if $f_j \notin F_{sel}$ then
12         Compute $I(f_{sel}; f_j)$.
13         Combine relevancy and redundancy as
           $\varnothing(f_j) = I(f_{sel};c) - \beta * I(f_{sel};f_j)$.
14         if $f_j \geq f_{j-1}$ then
15             $f_{sel} \leftarrow f_j$ [7]
16     Append $F_{\text{sel}} \leftarrow F_{\text{sel}} + f_{sel}$
17 Return $F_{\text{sel}}$
```

Algorithm 1: mRMR Feature Selection Algorithm

$S_{\text{tot}} = \{f_1, f_2, ...f_k\}$ of dimension k and a gesture g to mRMR algorithm, and receive a selected list of features F_{sel} with dimension m as an output, where $m \leq k$, and $F \subseteq S$. The subset F should produce high uniqueness and better classification accuracy compared to feature set S. The algorithm starts with an empty list and iteratively adds one feature at a time by keeping high relevancy and minimum redundancy. The relevancy is determined by measuring the mutual information of a feature with the class label $I(f_i;C)$ while redundancy is measured between features $I(f_i;f_j)$. The algorithm terminates when all features in S are exhausted. At the end, we obtain list of features F ranked merit-wise along with their MRMR values. In our case, we picked the features which were giving high relevancy, low redundancy, and where the improvement in mRMR values was not significant. The mRMR algorithm for finding the best subset of m features using forward selection strategy is formalized below.

Figure 8 shows resulting mRMR values corresponding to features of swipe, its sub-types, and handwriting. It is clear that mRMR values become a lot consistent after a certain range of features e.g. 170. Moreover, we observe a sharp decline in mRMR values after a set of 50 features which indicates that features have low relevancy to the class but high dependency to other features after a certain range. Also, note that x-axis starts with the second feature instead of first; the reason is that mRMR algorithm selects the first feature based only on the maximum relevancy between a feature and a class, as mentioned in Steps 4 to 7. While this could be a drawback of mRMR, we, however, validate selected features by applying our framework and also by using classification metrics.

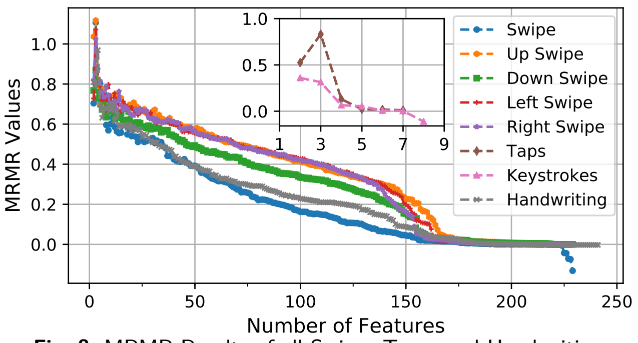


Fig. 8. MRMR Results of all Swipes Types and Handwriting

A.4 Users using Same Devices

We select "Nexus 5" as our primary device to analyze uniqueness results for users accessing our app through the same device. We chose "Nexus 5" because it is was most used model of phone in our study, primarily because our test smartphone is also Nexus 5, which were given to users who did not possess an Android phone, for data collection. Table 8 shows the statistics of analyzed data. Our results indicate that users are highly recognizable on the same device. For a set of gesture samples, the performance of keystrokes and taps are fairly better on a single device as compared to multiple devices. We found that features with a single data point,

6 mRMR follows filter-based approach with entropy and information gain being an inherent part of feature selection.

7 Incrementally select the jth feature from the remaining set $S - F_{sel}$ that maximizes $\varnothing(.)$ using the eq. $\mathbf{max}_{\mathbf{f_j \in S - F_{sel}}}[I(f_{sel}, c) - \beta * I(f_{sel}, f_j)]$

Table 8. Touch Data Statistics

Gesture	Users	Sp.	Gesture	Users	Sp.
Swipes	08	920	Up Swipes	08	244
Down Swipes	07	217	Left Swipes	07	214
Right Swipes	08	245	Handwriting	08	259
Taps	09	2653	Keystrokes	08	1614
Total Samples: 6366			Sp. = Samples		

such as Start X, Start Y, Start Pressure, Start Area, etc. highly contributes towards user uniqueness for keystroke and taps. Similarly, handwriting and overall swipes also show improved performance with the Finger Area being most prominent feature.

Figure 9 shows the CDF of a set of gesture samples. We observe that handwriting reflects 100% of user identification followed by swipes with 95.83% of mutual relative information. Keystrokes and taps reveal 85.5% and 77.7% of user information respectively. The performance of swipe sub-types are also improved except for the up swipes (54%).

Figure 10 is the CDF of a gesture sample. We found that the performance of a handwriting and a swipe is overall improved, with 100% and 80.8% of identification respectively. It is noted that these results reflect a subset of the user data for a single Nexus 5X device. It could be concluded that these results are influenced from the device type and size of the subset. In order to confidently verify our suspicions, we need to collect and analyze data from other types of devices with more user data and consider it as part of future work.

A.5 Results Summary

The ROC Curves of a gesture sample and set of gesture samples are shown in figure 11 and 12. Table 9 and 10 represent the summary of results corresponding to each gesture and combinations.

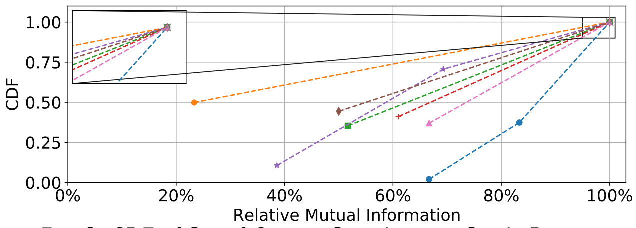


Fig. 9. CDF of Set of Gesture Samples on a Single Device. Relative Information of Respective Categories are: -●- Swipes: 95.8%, -⬡- Up Swipes: 54.0%, -■- Down Swipes: 80.9%, -+- Left Swipes: 83.7%, -★- Right Swipes: 66.0%, -◆- Taps: 77.7%, -▲- Keystrokes: 87.5%, -×- Handwriting: 100%

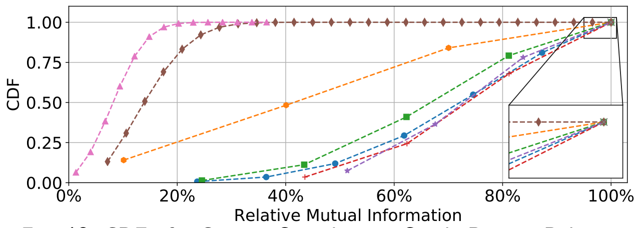


Fig. 10. CDF of a Gesture Sample on a Single Device. Relative Information of Respective Categories are: -●- Swipes: 80.8%, -⬡- Up Swipes: 52.4%, -■- Down Swipes: 77.9%, -+- Left Swipes: 87.1%, -★- Right Swipes: 82.0%, -◆- Taps: 14.4%, -▲- Keystrokes: 09.1%, -×- Handwriting: 100%

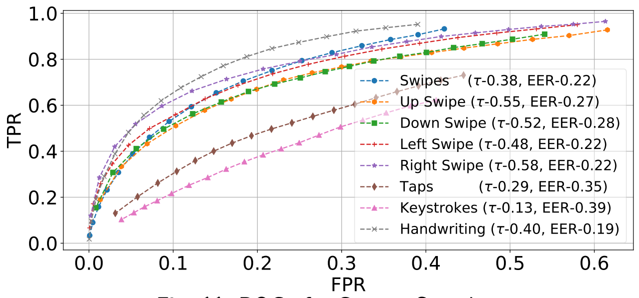


Fig. 11. ROC of a Gesture Sample

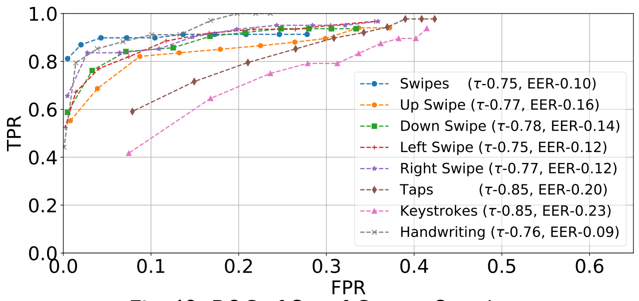


Fig. 12. ROC of Set of Gesture Samples

References

[1] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang. SilentSense: Silent User Identification via Dynamics of Touch and Movement Behavioral Biometrics. *MobiCom '13*, page 187, 2013.

[2] H. Bojinov and Y. Michalevsky. Mobile Device Identification via Sensor Fingerprinting. *arXiv preprint arXiv: . . .* , 2014.

[3] D. Chaffey. How many connected devices do consumers use today?. http://www.smartinsights.com/traffic-building-strategy/integrated-marketing-communications/many-connected-devices-use-today-chartoftheday/, 2016.

[4] T. Chen, A. Chaabane, P. U. Tournoux, M.-A. Kaafar, and R. Boreli. How much is too much? leveraging ads audience estimation to evaluate public profile uniqueness. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 225–244. Springer, 2013.

[5] M. Conti, I. Zachia-Zlatea, and B. Crispo. Mind how you answer me!: transparently authenticating the user of a smartphone when answering or placing a call. *Proceedings of the 6th ACM Symposium on Information, Computer*

Table 9. Summary of Results - Gesture Sample

Gesture	Gesture Sample		Set of Gesture Samples		Features List
	Rel. Inf.	TPR	Rel. Inf.	TPR	
Swipe	57.79%	76.11%	63.33%	89.85%	Stop Area, 80-percentile pairwise X-Tilt, Start Area, Mid-Stroke Pressure, 80-percentile pairwise Area, Std. Dev. of Pairwise Velocity, Std. Dev. of Pairwise Change of Area-Position, 20-percentile pairwise Area, 50-percentile pairwise Area, End to End Acc.*, Distance, Start Pressure, End Point of Pairwise Area, Start Point of Pairwise Area, Mid-Stroke Area, Std. Dev. of Pairwise Area, Average of Pairwise X-Tilt, Average of Pairwise Area, 50-percentile pairwise Pressure, 20-percentile pairwise Change of Area-Position, 80-percentile pairwise Pressure, Median of Last 3 Velocities Points, 20-percentile pairwise Pressure, Average of Pairwise Pressure, 80-percentile pairwise Change of Area-Position, Start Point of Pairwise Pressure, Stop Pressure, Start Point of Pairwise X-Tilt, Start Point of Pairwise Change of Area-Position, Std. Dev. of Pairwise Change of Pressure-Position, Start Point of Pairwise Velocity, End Point of Pairwise Pressure, 80-percentile pairwise Y-Tilt, Std. Dev. of Pairwise Pressure, Start Point of Pairwise Direction, Average of Pairwise Change of Area-Position, Start Y, 50-percentile pairwise X-Tilt, End to End Pressure Distance, 20-percentile pairwise X-Tilt, Start Point of Pairwise Change of Pressure-Position, Median of First 5 Acc. Points, Average of Pairwise Change of Pressure-Position, Start Point of Pairwise Y Velocity, Average Velocity, 20-percentile pairwise Change of Pressure-Position, Average of Pairwise Y-Tilt, 80-percentile pairwise Change of Pressure-Position, End Point of Pairwise X-Tilt, Direct End To End Direction, Average of Pairwise Y, Start Point of Pairwise Y-Tilt
Up Swipe	48.56%	74.11%	50.23%	84.44%	Stop Pressure, Std. Dev. of Pairwise X-Tilt, Average Velocity, Start Pressure, Start Y, Average of Pairwise X-Tilt, Std. Dev. of Pairwise Area, 20-percentile pairwise Pressure, 80-percentile pairwise Area, 20-percentile pairwise Area, Phone Orientation, End Point of Pairwise Pressure, Average of Pairwise Area, End Point of Pairwise X-Tilt, Stop Y, End Point of Pairwise Area, Start Point of Pairwise Pressure, Start X, Median of First 5 Acceleration Points, Average of Pairwise Pressure, 50-percentile pairwise Area Acc., 50-percentile pairwise Y Acc., 50-percentile pairwise Pressure Acc., 50-percentile pairwise Change of X-Tilt Position, Std. Dev. of Pairwise Pressure, Start Point of Pairwise X-Tilt, Start Point of Pairwise Area, Stop X, Average of Pairwise Direction, 80-percentile pairwise Pressure, Std. Dev. of Pairwise Y-Tilt, End to End Y Distance, 80-percentile pairwise Y Acc., 20-percentile pairwise X-Tilt, Length of Trajectory, 50-percentile pairwise Acc., 80-percentile pairwise Pressure Acc., Average of Pairwise Y-Tilt, 50-percentile pairwise X Acc., Std. Dev. of Pairwise X-Tilt Acc., 20-percentile pairwise Change of Area-Position, Start Point of Pairwise Y-Tilt, 20-percentile pairwise Direction, End to End X Distance, 50-percentile pairwise Area, End Point of Pairwise Y-Tilt, 50-percentile pairwise X, 20-percentile pairwise Change of Pressure-Position, Direct End To End Direction, 20-percentile pairwise Raw Y-Tilt
Down Swipe	52.22%	72.64%	54.58%	86.12%	Start Point of Pairwise Area, 50-percentile pairwise Acc., Median of First 5 Acc. Points, Start Y, Stop Pressure, Start Point of Pairwise Change of Area-Position, Start Pressure, 50-percentile pairwise Acc., 50-percentile pairwise Pressure Acc., Average Velocity, Std. Dev. of Pairwise Area, 80-percentile pairwise Pressure Acc., End Point of Pairwise Area, Average of Pairwise X-Tilt, Average of Pairwise Area, Start Point of Pairwise Pressure, Average of Pairwise Direction, 20-percentile pairwise Area, 20-percentile pairwise Pressure, 20-percentile pairwise Change of Area-Position, End Point of Pairwise Pressure, Start Point of Pairwise X-Tilt, 80-percentile pairwise Area, 50-percentile pairwise Y Acc., End to End Y Distance, Average of Pairwise Pressure, Start X, Stop Y, Average of Pairwise Change of Area-Position, Std. Dev. of Pairwise X-Tilt, Std. Dev. of Pairwise Pressure, Average of Pairwise Velocity, 80-percentile pairwise Y Acc., Ratio of End2End Dist. and Len of Trajectory, 80-percentile pairwise Pressure, 50-percentile pairwise Area, Average of Pairwise Y-Tilt, Average of Pairwise Y Velocity, End to End X Distance, Average of Pairwise X-Tilt Velocity, 20-percentile pairwise X-Tilt, 80-percentile pairwise Change of Y-Position, Start Area, 50-percentile pairwise Pressure, Start Point of Pairwise Y-Tilt, Start Point of Pairwise Change of Pressure-Position, 50-percentile pairwise Y, Std. Dev. of Pairwise X-Tilt Acc., Average of Pairwise Y, End Point of Pairwise X-Tilt
Left Swipe	53.96%	74.60%	68.66%	88.52%	20-percentile pairwise Area, Average of Pairwise X-Tilt, Stop Pressure, 80-percentile pairwise Area, Start Pressure, Average Velocity, 50-percentile pairwise Pressure Acc., Std. Dev. of Pairwise X-Tilt, Start Point of Pairwise Area, Median of First 5 Acc. Points, Start X, Std. Dev. of Pairwise Area, End Point of Pairwise Area, 80-percentile pairwise Pressure Acc., 50-percentile pairwise Acc., Average of Pairwise Area, 20-percentile pairwise Pressure, 20-percentile pairwise Change of Area-Position, 50-percentile pairwise Area Acc., End Point of Pairwise Pressure, Start Point of Pairwise Change of Area-Position, 50-percentile pairwise Y Acc., 80-percentile pairwise Area Acc., Start Y, 20-percentile pairwise X-Tilt, End to End Y Distance, Start Point of Pairwise Pressure, 50-percentile pairwise Change of X-Tilt Position, Average of Pairwise Pressure, Stop Y, Average of Pairwise Y-Tilt, 50-percentile pairwise Area, Std. Dev. of Pairwise Pressure, Start Point of Pairwise X-Tilt, 80-percentile pairwise Change of X-Tilt Position, 80-percentile pairwise Pressure, Average of Pairwise Change of Area-Position, Start Point of Pairwise Y, 50-percentile pairwise X-Tilt Acc., 80-percentile pairwise Y Acc., Average of Pairwise X-Tilt Velocity, Average of Pairwise Y Velocity, 20-percentile pairwise Y-Tilt, Average of Pairwise X-Tilt Acc., End Point of Pairwise X-Tilt, 20-percentile pairwise Change of Pressure-Position, 50-percentile pairwise X Acc., Start Point of Pairwise X-Tilt Velocity, Std. Dev. of Pairwise Y-Tilt, Average of Pairwise Velocity
Right Swipe	52.27%	76.37%	57.48%	86.24%	50-percentile pairwise Pressure Acc., Average of Pairwise X-Tilt, Stop Pressure, Start Pressure, 80-percentile pairwise Pressure Acc., Median of First 5 Acc. Points, Std. Dev. of Pairwise X-Tilt, Start Point of Pairwise Area, 20-percentile pairwise Area, Start Y, 80-percentile pairwise Area, End Point of Pairwise Area, Start X, Average of Pairwise Area, Std. Dev. of Pairwise Area, 50-percentile pairwise Area Acc., 20-percentile pairwise Pressure, Start Point of Pairwise Change of Area-Position, End Point of Pairwise Pressure, 80-percentile pairwise Change of X-Tilt Position, Average of Pairwise Direction, Average of Pairwise Pressure, Start Point of Pairwise X-Tilt, Average Velocity, End to End Y Distance, 20-percentile pairwise X-Tilt, 50-percentile pairwise Y Acc., Start Point of Pairwise Pressure, Average of Pairwise X-Tilt Velocity, 50-percentile pairwise Area, Average of Pairwise Y-Tilt, Stop Y, Std. Dev. of Pairwise Y-Tilt, 80-percentile pairwise Y Acc., 50-percentile pairwise Acc., 50-percentile pairwise X Acc., Std. Dev. of Pairwise Pressure, Average of Pairwise Change of Area-Position, 80-percentile pairwise Pressure, 80-percentile pairwise Area Acc., 50-percentile pairwise X-Tilt Acc., 50-percentile pairwise Change of X-Tilt Position, 20-percentile pairwise Y-Tilt, Average of Pairwise Velocity, 20-percentile pairwise X-Tilt Velocity, Start Point of Pairwise Y-Tilt, Start Point of Pairwise Y, End to End X Distance, End Point of Pairwise X-Tilt, Start Point of Pairwise Change of Pressure-Position
Keystroke	26.25%	60.00%	41.02%	75.00%	Inter-Stroke Time, Stroke Duration, Start X, Start Pressure, Start Y,Start Area, Mid-Stroke Finger Orientation, Key Error Rate
Tap	29.58%	63.33%	34.73%	79.54%	Inter-Stroke Time, Stroke Duration, Start X, Start Pressure, Start Y,Start Area, Mid-Stroke Finger Orientation
Handwriting	68.71%	81.16%	73.73%	91.11%	Average Pressure, Average Y, End Point of Pairwise X-Tilt, 80-percentile pairwise Pressure, 20-percentile pairwise Pressure, 80-percentile pairwise Area, Mid-Stroke Pressure, drawing width, 50-percentile pairwise Pressure, 80-percentile pairwise Y, Average of Pairwise Pressure, Std. Dev. of Pairwise X-Tilt, Start Pressure, Stop Pressure, End to End Y Distance, Start Point of Pairwise Pressure, End Point of Pairwise Y-Tilt, 20-percentile pairwise Y, End Point of Pairwise Pressure, 80-percentile pairwise X-Tilt, Std. Dev. of Pairwise Pressure, Start Point of Pairwise Direction, 50-percentile pairwise Y, 80-percentile pairwise X-Tilt Velocity, std. Dev. of Pairwise Change of Area-Position, TMP, Std. Dev. of Pairwise Change of Pressure-Position, Average of Pairwise Y, Std. Dev. of Pairwise X-Tilt Velocity, 20-percentile pairwise Area, Start Y, Std. Dev. of Pairwise Y-Tilt, 80-percentile pairwise Y-Tilt, drawing area, End to End Pressure Distance, 50-percentile pairwise Direction, 80-percentile pairwise Direction, 80-percentile pairwise Y-Tilt Velocity, Direct End To End Distance, Average of Pairwise X-Tilt, Start Point of Pairwise Y, Std. Dev. of Pairwise Y-Tilt Velocity, Stop Y, LMP, Stroke Duration, Start Point of Pairwise Change of Pressure-Position, 20-percentile pairwise Direction, 50-percentile pairwise X-Tilt, 80-percentile pairwise Change of X-Tilt Position, 20-percentile pairwise Change of Area-Position

* Acc. refers to Acceleration.

Table 10. Summary of Results - Gestures Combinations

Gesture	Rel. Inf.	TPR	FPR	Gesture Combinations	Rel. Inf.	TPR	FPR
Swipes, Taps	48.11%	72.72%	16.75%	Swipes, Keystrokes	46.80%	91.10%	22.40%
Swipes, Handwriting	72.75%	93.75%	10.88%	Taps, Keystrokes	33.55%	91.11%	33.83%
Taps, Handwriting	66.31%	88.57%	12.68%	Keystrokes, Handwriting	68.26%	72.41%	13.17%
Swipes, Taps, Keystrokes	93.87%	39.4%	2.3%	Swipes, Taps, Handwriting	96.10%	68.75%	1.8%
Swipes, Keystrokes, Handwriting	98.54%	51.85%	0.85%	Taps, Keystrokes, Handwriting	95.06%	51.72%	2.46%
All Gestures	98.93%	40.74%	0.99%				

and Communications Security, pages 249–259, 2011.

[6] J. Corripio, D. González, A. Orozco, L. Villalba, J. Hernandez-Castro, and S. Gibson. Source smartphone identification using sensor pattern noise and wavelet transform. *5th International Conference on Imaging for Crime Detection and Prevention, ICDP 2013*, 2013.

[7] A. Das and N. Borisov. Poster : Fingerprinting Smartphones Through Speaker. *35th IEEE Symposium on Security and Provacy*, pages 2–3, 2014.

[8] A. Das, N. Borisov, and M. Caesar. Do You Hear What I Hear?: Fingerprinting Smart Devices Through Embedded Acoustic Components. *Ccs*, pages 441–452, 2014.

[9] A. Das, N. Borisov, and M. Caesar. Tracking Mobile Web Users Through Motion Sensors : Attacks and Defenses. *Ndss*, (February):21–24, 2016.

[10] C. De Boor. *A practical guide to splines*, volume 27 of *Applied mathematical sciences*. Springer-Verlag New York, 1978.

[11] M. O. Derawi, C. Nickely, P. Bours, and C. Busch. Unobtrusive user-authentication on mobile phones using biometric gait recognition. *Proceedings - 2010 6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IIHMSP 2010*, pages 306–311, 2010.

[12] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi. AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable. *Network and Distributed System Security Symposium (NDSS)*, (February):23–26, 2014.

[13] P. Eckersley. How Unique Is Your Browser? *Proc. of the Privacy Enhancing Technologies Symposium (PETS)*, pages 1–18, 2010.

[14] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Transactions on Information Forensics and Security*,

8(1):136–148, 2013.

[15] C. Giuffrida, K. Majdanik, M. Conti, and H. Bos. I sensed it was you: Authenticating mobile users with sensor-enhanced keystroke dynamics. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8550 LNCS:92–111, 2014.

[16] M. Jakobsson, E. Shi, P. Golle, and R. Chow. Implicit authentication for mobile devices. *Proceedings of the 4th USENIX conference on Hot topics in security (HotSec'09)*, page 9, 2009.

[17] P. Kang and S. Cho. Keystroke dynamics-based user authentication using long and free text strings from various input devices. *Information Sciences*, 308:72–93, 2015.

[18] T. Kohno, A. Broido, and K. C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, 2005.

[19] A. Kurtz, H. Gascon, T. Becker, K. Rieck, and F. Freiling. Fingerprinting Mobile Devices Using Personalized Configurations. *Proceedings on Privacy Enhancing Technologies*, 2016(1):4–19, 2016.

[20] P. Laperdrix, W. Rudametkin, and B. Baudry. Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints. *Proceedings - 2016 IEEE Symposium on Security and Privacy, SP 2016*, pages 878–894, 2016.

[21] E. Maiorana, P. Campisi, N. González-Carballo, and A. Neri. Keystroke dynamics authentication for mobile phones. *Proceedings of the 2011 ACM Symposium on Applied Computing SAC 11*, pages 21–26, 2011.

[22] J. R. Mayer. Internet Anonymity in the Age of Web 2.0. *A Senior Thesis presented to the Faculty of the Woodrow Wilson School of Public and International Affairs in partial fulfillment of the requirements for the degree of Bachelor of Arts.*, page 103, 2009.

[23] Y. Meng, D. S. Wong, R. Schlegel, and L. F. Kwok. Touch gestures based biometric authentication scheme for touchscreen mobile phones. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7763 LNCS:331–350, 2013.

[24] Ł. Olejnik, G. Acar, C. Castelluccia, and C. Diaz. The leaking battery: A privacy analysis of the HTML5 battery status API. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9481:254–263, 2016.

[25] Ł. Olejnik, C. Castelluccia, and A. Janc. Why Johnny Can't Browse in Peace: On the Uniqueness of Web Browsing History Patterns. *5th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs 2012)*, pages 1–16, 2012.

[26] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[27] D. Perito, C. Castelluccia, M. A. Kaafar, and P. Manils. How unique and traceable are usernames? In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 1–17. Springer, 2011.

[28] N. Sae-bae, N. Memon, K. Isbister, and K. Ahmed. Multitouch Gesture-Based Authentication can the system accurately distinguish between. 9(4):568–582, 2014.

[29] D. W. Scott. On optimal and data-based histograms. *Biometrika*, 66:605–610, 1979.

[30] S. Seneviratne, A. Seneviratne, P. Mohapatra, and A. Mahanti. Predicting user traits from a snapshot of apps installed on a smartphone. *Mobile Computing and Communications Review*, 18(2):1–8, 2014.

[31] M. Shahzad, A. X. Liu, and A. Samuel. Secure Unlocking of Mobile Touch Screen Devices by Simple Gestures – You can see it but you can not do it. *Proc. of MobiCom*, page 39, 2013.

[32] M. Sherman, G. Clark, Y. Yang, S. Sugrim, A. Modig, J. Lindqvist, A. Oulasvirta, and T. Roos. User-generated free-form gestures for authentication: Security and memorability. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*, pages 176–189. ACM, 2014.

[33] E. Shi, Y. Niu, M. Jakobsson, and R. Chow. Implicit authentication through learning user behavior. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6531 LNCS:99–113, 2011.

[34] L. Sweeney. Simple demographics often identify people uniquely. *Carnegie Mellon University, Data Privacy Working Paper 3. Pittsburgh 2000*, pages 1–34, 2000.

[35] M. Tamviruzzaman, S. I. Ahamed, C. S. Hasan, and C. O'brien. ePet:when cellular phone learns to recognize its owner. *Proceedings of the 2nd ACM workshop on Assurable and usable security configuration - SafeConfig '09*, page 13, 2009.

[36] C. M. Tey, P. Gupta, and D. Gao. I can be You: Questioning the use of Keystroke Dynamics as Biometrics. *20th Annual Network and Distributed System Security Symposium - NDSS '13*, pages 1 – 16, 2013.

[37] H. Xu, Y. Zhou, and M. R. Lyu. Towards Continuous and Passive Authentication via Touch Biometrics: An Experimental Study on Smartphones. *SOUPS '14: Proceedings of the Tenth Symposium On Usable Privacy and Security*, pages 187–198, 2014.

[38] T.-F. Yen, Y. Xie, F. Yu, R. P. Yu, and M. Abadi. Host Fingerprinting and Tracking on the Web: Privacy and Security Implications. *Network and Distributed System Security Symposium*, pages 1–16, 2012.

[39] S. Zahid, M. Shahzad, S. A. Khayam, and M. Farooq. Keystroke-based User Identification on Smart Phones.pdf. pages 1–18.

[40] X. Zhao, T. Feng, and W. Shi. Continuous mobile authentication using a novel Graphic Touch Gesture Feature. *IEEE 6th International Conference on Biometrics: Theory, Applications and Systems, BTAS 2013*, 2013.

[41] N. Zheng, K. Bai, H. Huang, and H. Wang. You are how you touch: User verification on smartphones via tapping behaviors. *Proceedings - International Conference on Network Protocols, ICNP*, pages 221–232, 2014.

[42] Z. Zhou, W. Diao, X. Liu, and K. Zhang. Acoustic Fingerprinting Revisited: Generate Stable Device ID Stealthily with Inaudible Sound. *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security - CCS '14*, pages 429–440, 2014.