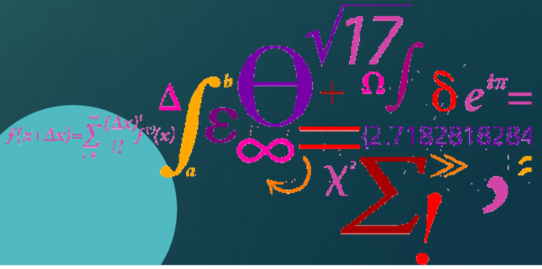
Towards Touch Behavioral Authentication on Mobile Devices: Design and Open Challenges

Weizhi Meng

Associate Professor

Department of Applied Mathematics and Computer Science,
Technical University of Denmark, Denmark



SECURWARE - November 2021

Research Directions

Weizhi Meng
weme@dtu.dk

http://www.staff.dtu.dk/weme

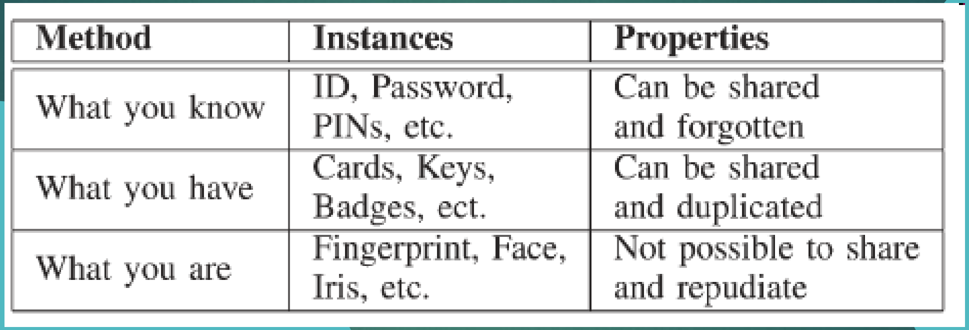
- Intrusion Detection
- Biometric Authentication
- Trust Computation
- HCI Security (including Graphical Password)
- Smartphone Security
- Blockchain

Outline

- Do We Need Biometric Authentication
- State-of-the-Art & Feasibility
- Reality and Behavioral Deviation
- How to Design a Robust Touch Behavioral Authentication

User Authentication on Mobile Devices

Existing User Authentication Methodologies.



Method	Instances	Properties
What you know	ID, Password, PINs, etc.	Can be shared and forgotten
What you have	Cards, Keys, Badges, ect.	Can be shared and duplicated
What you are	Fingerprint, Face, Iris, etc.	Not possible to share and repudiate

- Currently, the most widely used techniques of user authentication are the **passwords** (or passcodes) and **PINs** (personal identification numbers)

- ***Limitations:***
- External. Malware, shoulder surfing attack, etc.
- Internal. Long-term memory limitation, etc.

Biometric Authentication

- To overcome the drawbacks of passwords and PINs based authentication, research is being done into biometrics-based methods for authenticating users on mobile phones, as ***biometric characteristics can be unique and not duplicable or transferable***.

- ***Biometrics.*** An automated method of authentication by using measurable and enduring human physiological or behavioral characteristics to model and represent a user's identity.

- **Physiological** and **behavioral** approaches.
  - Use measurements from the human body such as fingerprint recognition, face recognition, iris recognition, retina recognition, etc..
  - Use measurements from human actions such as voice recognition, signature recognition, keystroke dynamics, touch dynamics, etc.

Do we need biometric authentication?

Yes

How about behavioral authentication?

Motivation

We need behavioral authentication!?

There are many applications of physilogical authentication, but why fewer applications of behavioral authentication!

Behavioral biometrics (1)

- **Voice Recognition**: This biometric attempts to identify a person who is speaking by characterizing his/her voice. The key point is that each human has different voice signatures, and identical words may have different meanings if spoken with different inflections or in different contexts.

- **Signature Recognition**: This technique measures and analyzes the physical activity of signing, while the core of a signature biometric system is behavioral. Traditionally, there are two ways to perform this recognition: static (i.e., signing on a paper) and dynamic (i.e., signing on a digitizing tablet). In the context of mobile phones, signature recognition is assumed to be dynamic, in which users should write their signatures in a digitizing tablet and in real-time.

- **Gait Recognition**: This type of recognition techniques is an emerging biometric technology which involves people being identified purely through the analysis of the way they walk. Currently, this kind of biometrics is still under development while it is feasible to be deployed on mobile phones as most phones like iPhones now can provide accelerometers with three primary axes (x, y, z).

Behavioral biometrics (2)

- **Behavior Profiling**: This kind of techniques aims to identify people based upon the way in which they interact with the services of their mobile devices. During the authentication, current users' activities such as dialling a telephone number are compared with an existing profile (which is built from historical usage) through a machine learning method.

- **Keystroke Dynamics**: This dynamics utilizes the manner and the rhythm of an individual when typing characters on a keyboard or keypad. It was well-known and has been studied for a long time in authenticating users on mobile devices.

- **Touch dynamics**. With the rapid development of mobile platforms, touchscreens have recently become a leading input method, which are an electronic visual display that users can control through simple or multi-touch gestures by touching the screen. touch dynamics, which refers to collecting detailed information about individual touches such as touch duration and touch direction, has become very popular in mobile market and is an emerging hot topic in literature.

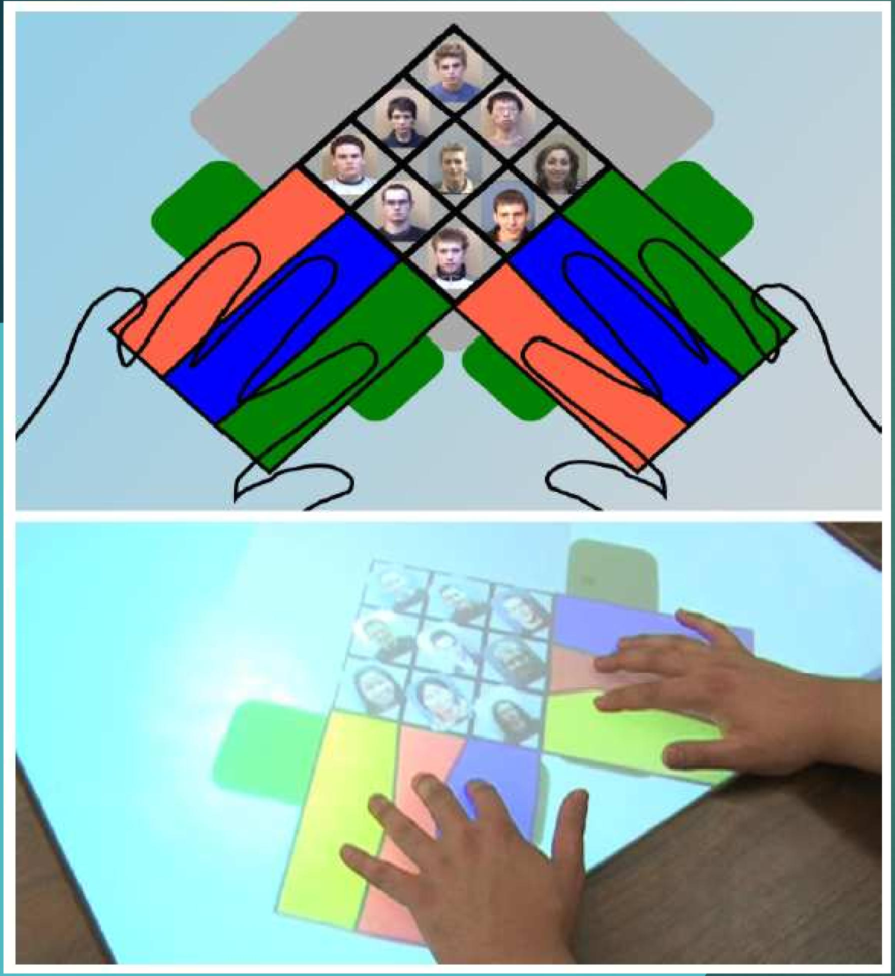
The Need of Touch Behavioral Authentication on Smartphones

- Depends on scenarios – the security requirements
- Depends on authentication performance – accuracy
- Depends on authentication usability and stability

Outline

- Do We Need Biometric Authentication
- State-of-the-Art & Feasibility
- Reality and Behavioral Deviation
- How to Design a Robust Touch Behavioral Authentication

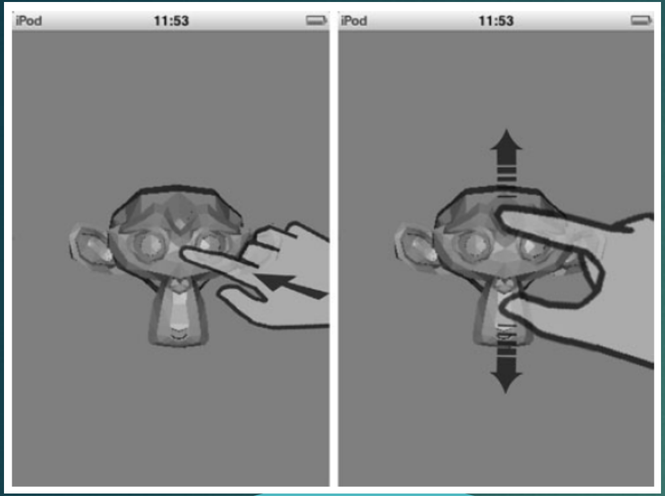
- The idea of using touch behavior for user authentication is not new, but most of the previous research focuses on desktop machines or on finger identification in 2009.



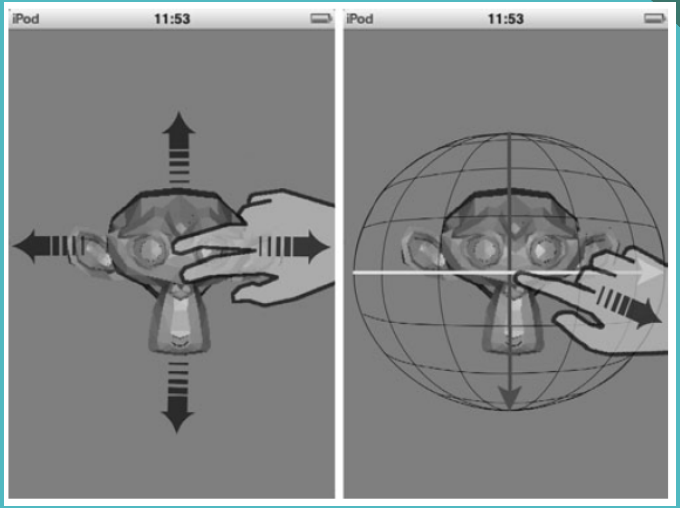
- Kim et al. in 2010 exploited the features of multi-touch interaction to inhibit shoulder surfing at an ATM. In their study, the user begins by placing three fingers of each hand in calibration areas on the interface. The system uses the locations of these touch points to dynamically draw the grid of objects, and pressure zones that are assigned to each finger.

- Picture (A) shows their proposed system. The user increases pressure on one finger per hand in the colored pressure zones to communicate an (x, y) coordinate and select an object.

Kim, D., Dunphy, P., Briggs, P., Hook, J., Nicholson, J.W., Nicholson, J., Olivier, P.: Multi-Touch Authentication on Tabletops. In: Proceedings of the 28th International Conference on Human Factors in Computing Systems (CHI), pp. 1093–1102, ACM, New York, USA (2010)



Picture (B)



Picture (C)

▪Fiorella et al. provided an evaluation of multi-touch input (e.g., rotation, translation, and scaling) for 3D object manipulation on mobile devices and compared their proposed Multi-touch user interface to a traditional button GUI. In their statistic analysis of 27 users on an iPod-touch device, they found that the Multi-touch user interface outperformed the traditional button GUI.

•Picture (B) shows the designed multi-touch interface, with picking (left) and scaling (right).

•Picture (C) shows the operations of translation (left) and rotation (right).

Fiorella, D., Sanna, A., Lamberti, F.: Multi-touch User Interface Evaluation for 3D Object Manipulation on Mobile Devices. Journal on Multimodal User Interfaces 4(1), 3–10 (2010)

Current Research

There is no big breakthrough on touch behavioral authentication in the past years

From a different angle, we next introduce two fundamental work in this area.

2012/2013

Y. Meng, D.S. Wong, R. Schlegel, and L.F. Kwok, 'Touch Gestures Based Biometric Authentication Scheme for Touchscreen Mobile Phones,` In: Proc. of the 8th China International Conference on Information Security and Cryptology (INSCRYPT), pp. 331-350, LNCS, Springer, Heidelberg, 2012.

- In this work, we develop a user authentication system based on touch dynamics, including 21 touch gesture-based features.

Touch dynamics (1)

- Touch dynamics: characteristics of the inputs received from a touchscreen when a user is interacting with a device.

- **Differences between Touch, Keystroke Dynamics and Mouse Dynamics:**
- Touch dynamics is different from keystroke dynamics in that touch dynamics has more input types such as multi-touch and touch-movement.
- Keystroke dynamics only has buttons as input devices, which do not have a movement feature, while touch dynamics has movement and can therefore provide more behavioral characteristics.

- Touch dynamics is also different from mouse dynamics in that touch dynamics has a possibility of multi-touch input.
- Looking at mouse dynamics, the trace in mouse dynamics is continuous (i.e., mouse inputs start from the last point where the last mouse input was terminated) while the trace in touch dynamics can be non-continuous (i.e., a touch input can start at a different point than the point where the last touch input ended).

Touch dynamics (2)

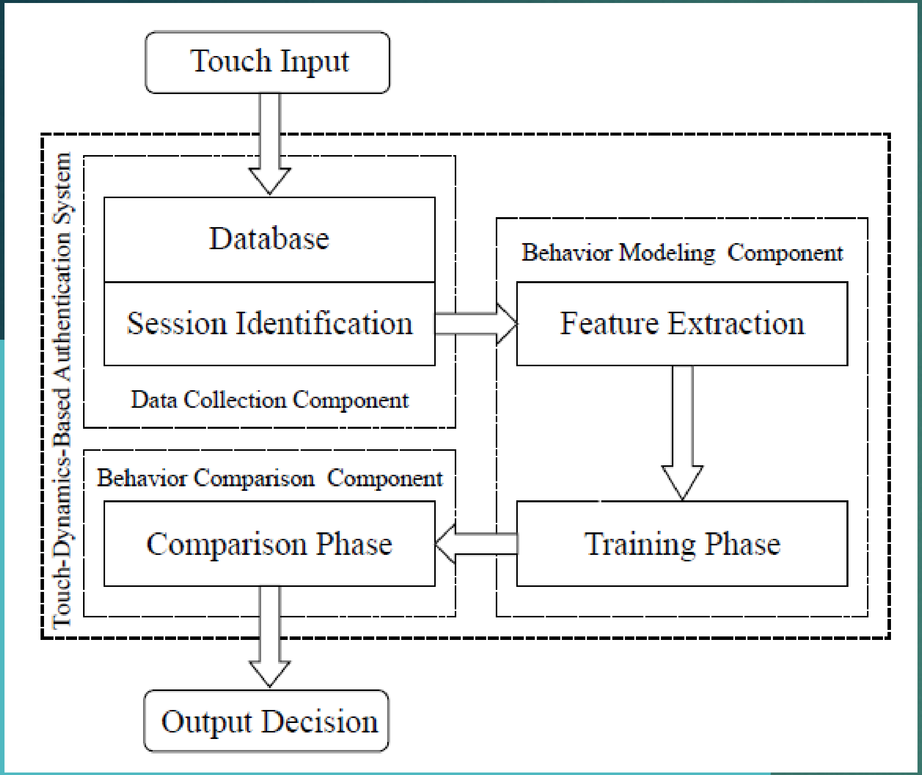
- **Similarities between Touch, Keystroke and Mouse Dynamics.**

- The inputs of press button up and press button down in keystroke dynamics are similar to the actions of touch press up and touch press down (e.g., single touch) in touch dynamics.
- Compared to mouse dynamics, touch dynamics has similar movement input types (i.e., mouse movement versus touch movement). In addition, a single touch input can be considered to be similar to a click action in mouse dynamics. Touch dynamics can therefore be considered as a combination of keystroke dynamics and mouse dynamics with respect to the main input types.
- This allows to use some behavioral features in touch dynamics that are also used in keystroke dynamics and mouse dynamics.

Touch dynamics (3)

- In this paper, we classify inputs as captured by the touchscreen on a mobile phone into four categories:
  - Single-Touch (ST): the input starts with a touch press down, followed by a touch press up without any movement in-between.
  - Touch-Movement (TM): the input starts with a touch press down, movement (also called drag), followed by a touch press up.
  - Multi-Touch (MT): an input with two or more simultaneous, distinct touch press down events at different coordinates of the touch screen (i.e., two fingers press down on the touchscreen simultaneously), either with or without any movement before a touch press up event.
  - No input: there is no input on the touchscreen.

Architecture (1)



Touch Input
Touch-Dynamics-Based Authentication System
Database
Session Identification
Data Collection Component
Behavior Modeling Component
Feature Extraction
Training Phase
Behavior Comparison Component
Comparison Phase
Output Decision

Figure shows the architecture of the touch-dynamics-based authentication system.

- **Data collection**: collects raw data from the touchscreen (i.e., recording and storing all touch gesture data into a database) and converting the raw data into meaningful information (i.e., identifying sessions).
- **Behavior modeling**: analyzes collected data, extracts features to generate authentication signature for a legitimate user, models a user's touch behavior.
- **Behavior comparison**: compares the current user's behavior with the relevant generated authentication signatures, and makes an output.

Architecture (2)

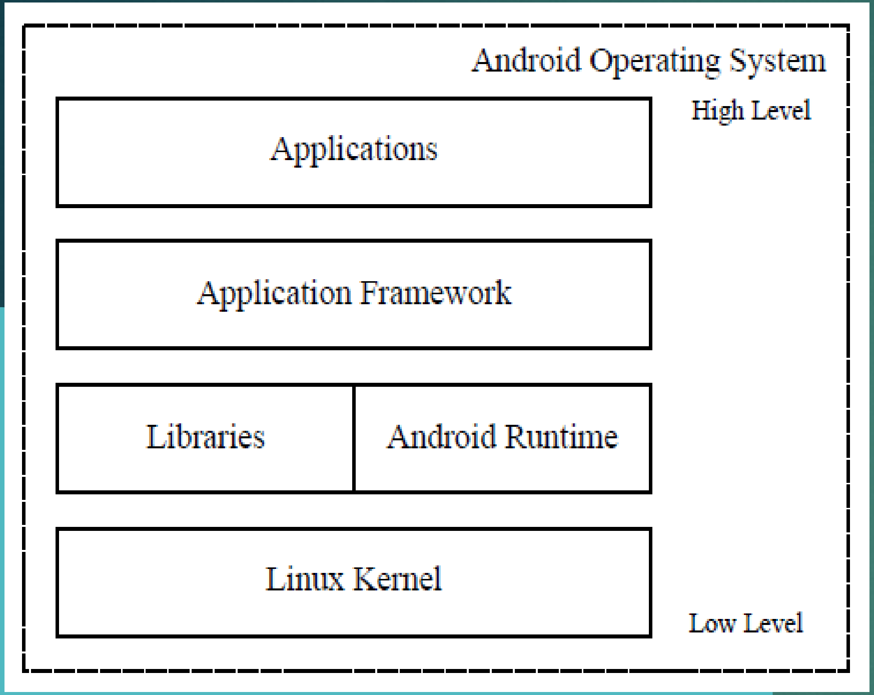
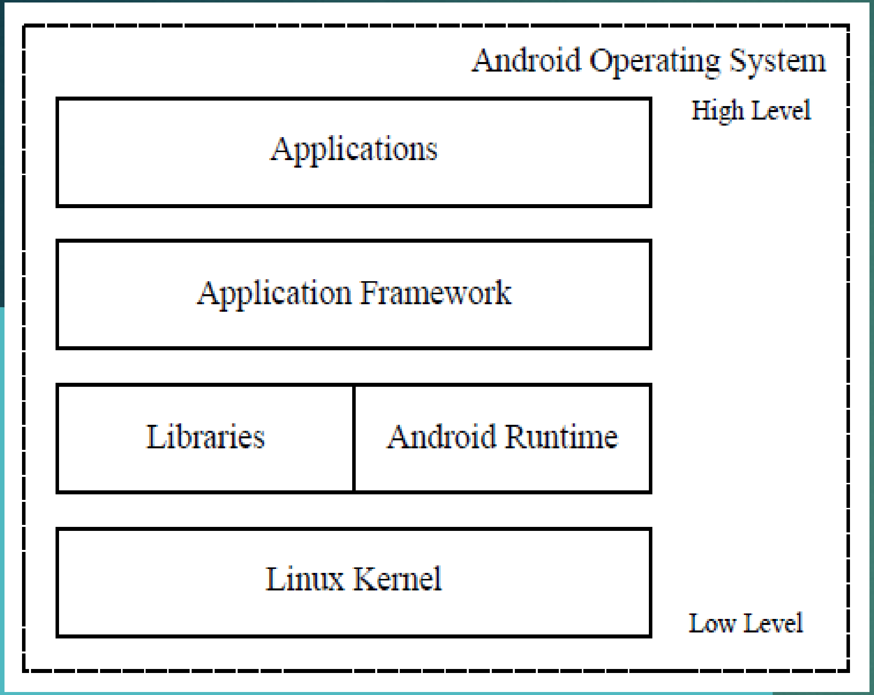


Figure . The architecture of the android operating system and its layers.

- **Linux kernel**. Android relies on Linux version 2.6 for core system services such as security, memory management and drivers. This layer contains drivers for devices such as USB, display, camera, Bluetooth chip and flash memory. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.
- **Libraries**. Android includes a set of C/C++ libraries such as the System C library, media libraries and 3D libraries, which are all used by various components of the Android system.
- **Android runtime**. Android includes a runtime which contains a set of core libraries that provide different functionalities. In addition, every Android application runs in its own process, with its own virtual machine instance.

Architecture (3)



- **Application framework**. The Android application framework is a high-level layer to provide the developer with a development platform for creating new Android applications. Developers can access location information, run background services, add notifications to the status bar, and access lots of other information and functionality.
- **Applications**. This is the highest level of the Android operating system architecture. Android ships with a set of core applications and widgets including an email client, messaging application, calendar, maps, browser, contacts and others. Users can also easily add more applications.

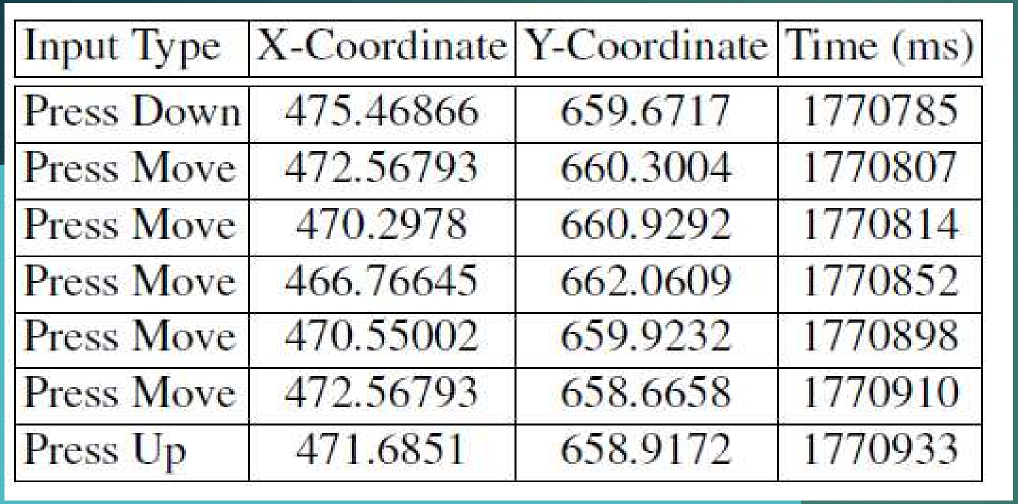
In our case, modifying the application framework layer allows us to implement the desired functionality without the need to modify any applications, and it is more applicable to develop a system by programming the application framework as an interface is provided by Android.

Data Collection (1)

- We used a Google/HTC Nexus One Android phone (CPU: 1GHz, Memory: 512 MB) with a capacitive touchscreen (resolution 480X800 px) to perform the experiments.
- The advantage of this particular phone is that the stock Android operating system installed on it can be replaced with a modified custom version of the Android OS. In particular, we updated the phone with a modified Android OS version 2.2 based on CyanogenMod.
- The modification consists of changes to the application framework layer to record raw input data from the touchscreen, such as the timing of touch inputs, the coordinates x and y, and the type of the input (e.g., single-touch, multi-touch or movement).
- In addition, we installed a separate application, which allowed us to easily extract the recorded data from the phone.

Data Collection (2)

Table gives a sample of raw data collected from touchscreen inputs.



Input Type	X-Coordinate	Y-Coordinate	Time (ms)
Press Down	475.46866	659.6717	1770785
Press Move	472.56793	660.3004	1770807
Press Move	470.2978	660.9292	1770814
Press Move	466.76645	662.0609	1770852
Press Move	470.55002	659.9232	1770898
Press Move	472.56793	658.6658	1770910
Press Up	471.6851	658.9172	1770933

- Each record consists of at least the following four fields: input type, x-coordinate, y-coordinate, and system time (S-time).
- The system time in Table 1 is relative to the last start-up of the phone.
- The duration of each touch input can then be calculated by taking the difference in system-time.
- These four fields allow us to precisely determine the type of touch inputs, their coordinates and their duration.

Data Collection (3)

- Session identification: the purpose is to determine when a new session starts.
- The specific length of a session can be configured.
  - A new session starts when a touch input is recorded and the last session has ended.
  - A session ends if the duration of the current session has reached or exceeded the maximum session duration time. For instance, if we choose a session duration time of 10 minutes, then our scheme will terminate a session and start a new session when the duration time of the current session reaches or exceeds 10 minutes.

Data Collection (4)

- In this work, we set the session length to 10 minutes by considering both user’s and system’s requirements.
  - For the user, a shorter session is desirable.
  - For the system, a longer session can provide more information to better model a user’s behavior.
  - In our current work, we consider an accurate user model is more important and the session length of 10 minutes is widely accepted by our participants, thus, we set the session length to 10 minutes. (We may evaluate other values in future work.)

Feature Extraction (1)

- In this work, we extract 21 features to construct an authentication signature for user authentication.
- The features are the following:

1. Average touch movement speed per direction (8 directions)
2. Fraction of touch movements per direction (8 directions)
3. Average single-touch time
4. Average multi-touch time
5. The number of touch movements per session
6. The number of single-touch events per session
7. The number of multi-touch events per session.

Feature Extraction (2)

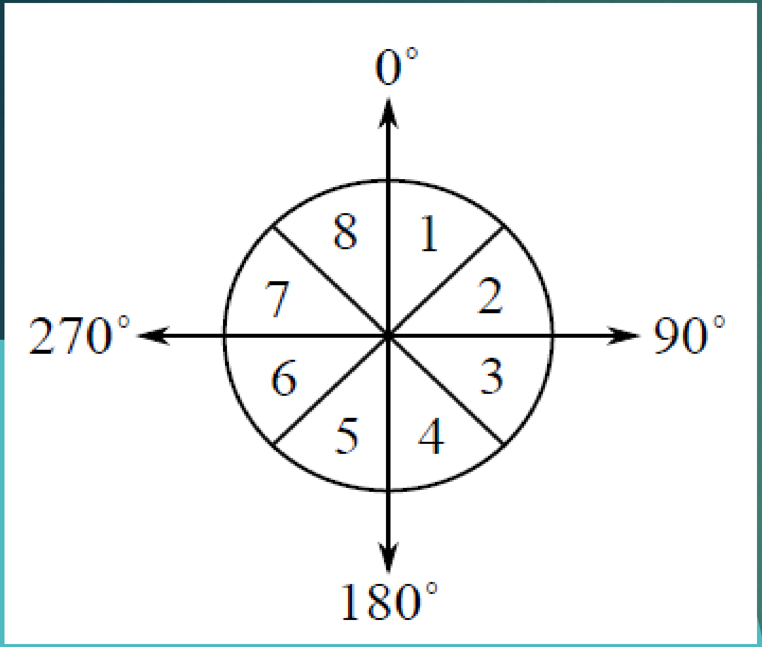


Figure shows the 8 different directions of a touch movement.

- **Average Touch Movement Speed per Direction**:

- After categorizing the touch movements according to their direction, we then calculate the average touch movement speed (denoted ATMS) for each of the 8 directions, represented by *ATMSi* (e.g., ATMS1 represents the ATMS in direction 1, ATMS3 represents the ATMS in direction 3).

- Touch movement speed (TMS):

$$TMS = \frac{\sqrt{(x2 - x1)^2 + (y2 - y1)^2}}{S2 - S1}$$

- Touch movement angle:

$$Touch\ movement\ angle:\ \theta = \arctan\frac{y2 - y1}{x2 - x1}, \theta \in [0, 360^\circ]$$

Feature Extraction (3)

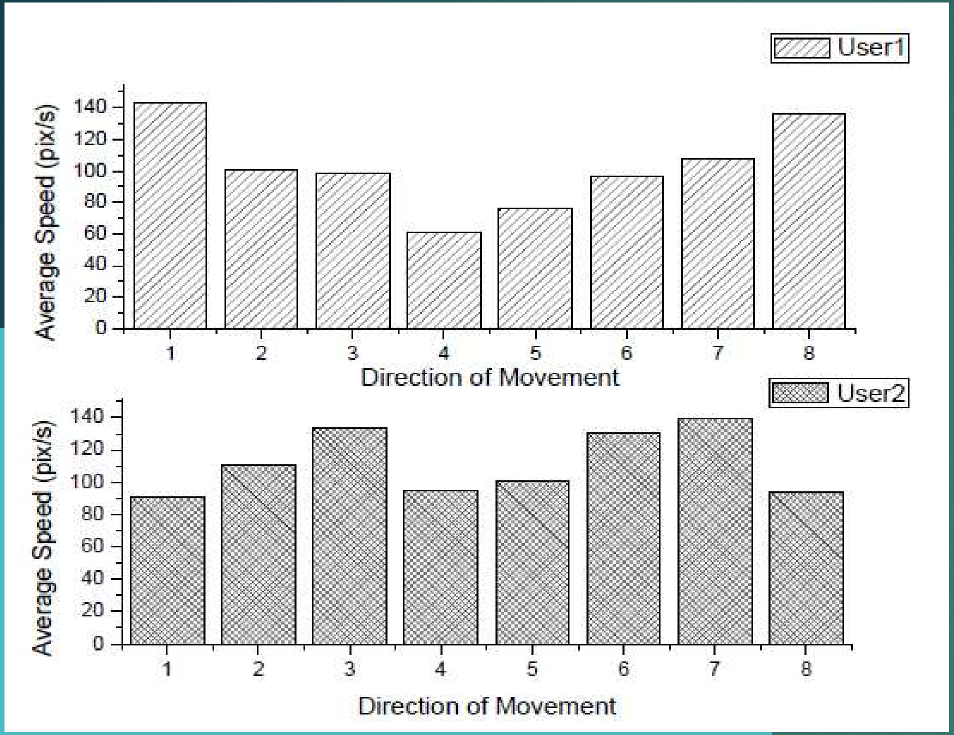


Figure shows the average touch movement speed versus the direction of movement for 2 different users.

• It is clearly visible that the distributions for these two users are different: the touch movements of User1 in direction 1 and 8 are performed with a higher speed than other directions, while the touch movements of User2 have a higher speed in direction 2, 3, 6, and 7. This illustrates nicely that the feature ATMS per direction (total of 8 features) can be used to model the characteristics of a user’s touch behavior.

Feature Extraction (4)

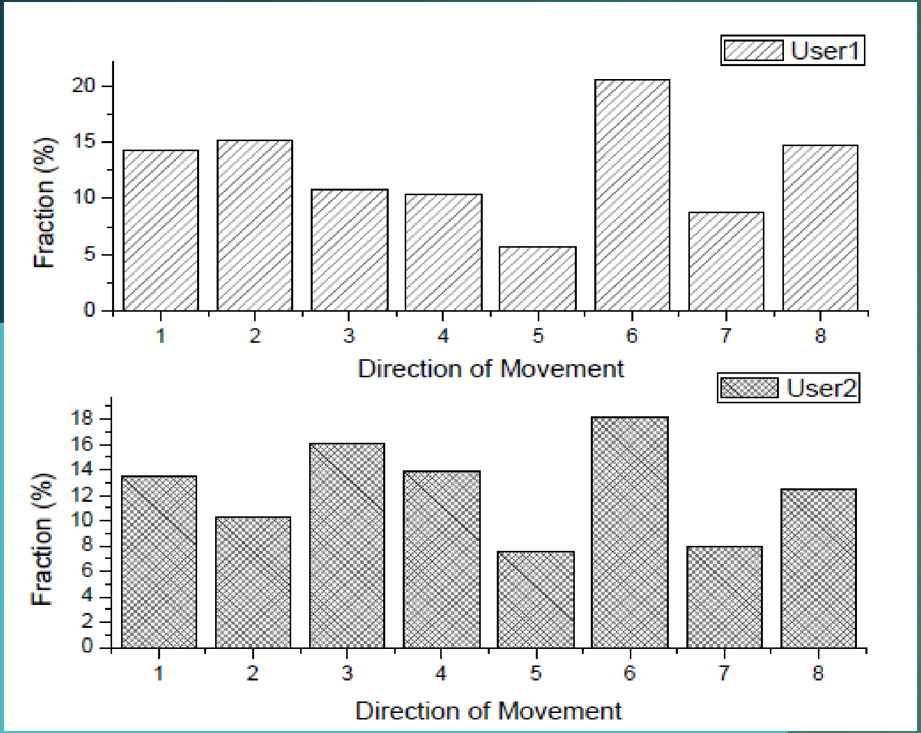


Figure shows the fraction of touch movements versus the direction of movement for 2 different users.

- Fraction of Touch Movements per Direction (**FTM**)
- We observe that there are usually certain directions that contain more touch movements than other directions and that for different users the fraction per direction varies.
- It shows the distribution of the fractions of touch movements (denoted FTM) versus the direction of a touch movement for User1 and User2.
- User1 performed relatively more touch movements in direction 1, 2, 6 and 8, while User2 performed more touch movements in direction 1, 3, 4, 6, and 8.
- The FTM in 8 directions (total of 8 features) can be used to characterize the touch behavior of a user.

Feature Extraction (5)

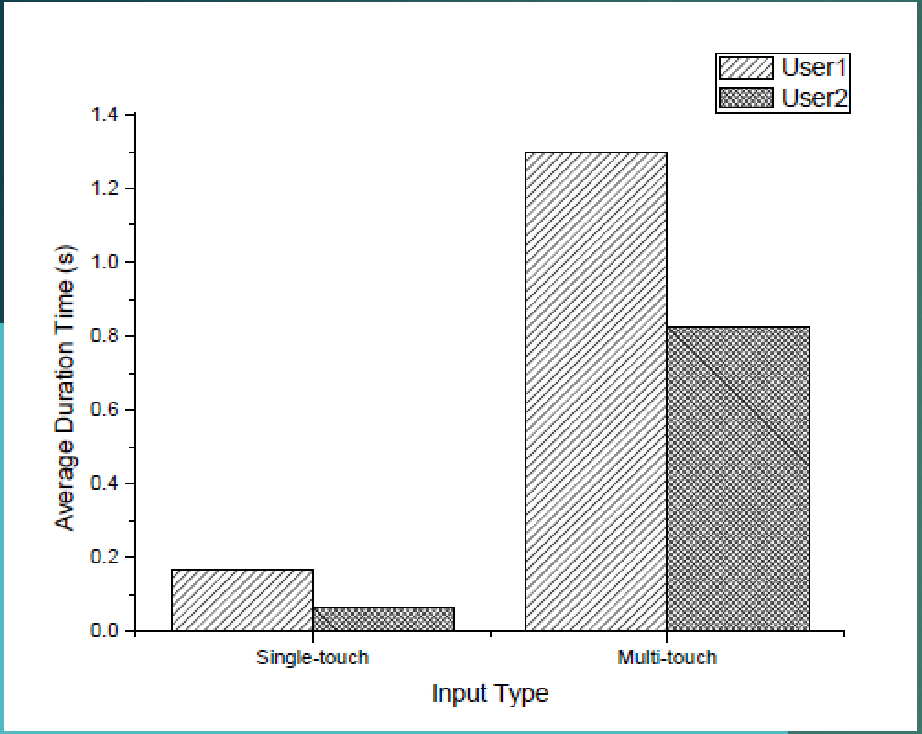


Figure shows the average single-touch time and the average multi-touch time for 2 different users.

- Average Single-touch/Multi-touch Time (**AST/MTT**)
- In addition to touch movements, single-touch and multi-touch are also two important types of touch inputs. We observe that the average duration time of a single-touch or multi-touch is different for different users.
- It shows the histogram for these two features, Average Single-touch time (denoted AST) and Average Multi-touch time (denoted MTT) again for the two users User1 and User2.
- User1 on average spent a longer time for AST and MTT compared to User2, showing that these two features can also be used to characterize and hence distinguish the touch behavior of different users.

Feature Extraction (6)

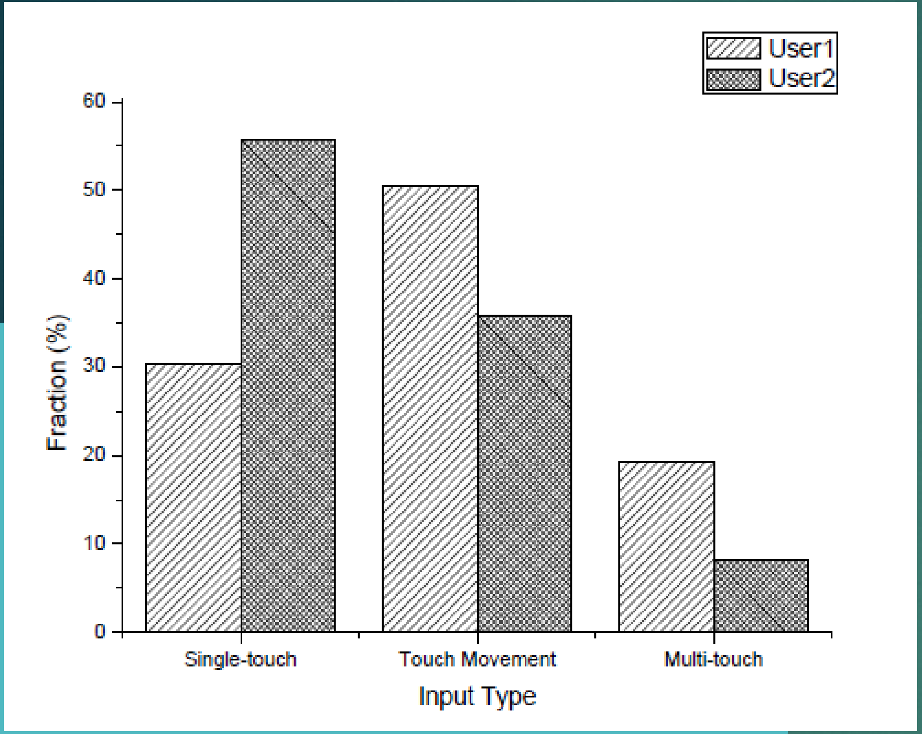


Figure shows the number of single-touch events, touch movements and multi-touch events per session for 2 different users.

- Number of Touch Action Events (**AST/MTT**)
- Single-touch, touch movement and multi-touch events are three major input types on a touchscreen.
- we observe that the total number of these three touch events over one session varies for different users.
- We therefore distinguish the three features number of touch movements per session (denoted NTM), number of single-touch events per session (denoted NSTE), and number of multi-touch events per session (denoted NMTE).

We can find that User1 performed more touch movements and multi-touches than User2, while User2 performed more single-touches than User1. It is also clearly visible that the numbers differ significantly between the users,

Training and Comparison

- In the **training phase** of the behavior modeling component, our scheme uses a classifier to recognize a user's profile by training with the user's authentication signatures. The training itself can be further divided into two types: initial training and dynamic training. A training phase starts with the initial training by collecting and utilizing several initial sessions from a user (i.e., several authentication signatures) to model a user's profile. Then it moves to dynamic training, which continuously trains the authentication system to integrate changes in the user's behavior.

- In the comparison phase of the behavior comparison component, the system extracts the authentication signature from the current user's touch behavior and compares it with the profile of a legitimate user.

Evaluation (1)

- We investigate the performance of 5 existing classification schemes when applied to our system: Decision tree (J48), Naive Bayes, Kstar, Radial Basis Function Network (RBFN) and Back Propagation Neural Network (BPNN).

- **J48** is a decision tree classifier that classifies data items by generating decision trees from training data.
- **Naive Bayes** is a probabilistic classifier based on the assumption that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature.
- **Kstar** is a statistical classifier based on the assumption that similar instances will have similar classes. Thus, it uses similarity functions to create instance-based classifications.
- **RBFN** and **BPNN** are neural network classifiers. RBFN is an artificial neural network that uses radial basis functions as activation functions. Its approximation capabilities are used to model complex mappings. The BPNN classifier has two main steps: (1) to present input and propagate it forward through the network to compute the output values for each output unit; (2) to perform backward passes through the network and calculate appropriate weights.

Evaluation (2)

- To remove any implementation related bias, we performed our evaluation using WEKA (using default settings), which is an open-source machine learning software that provides a collection of machine learning algorithms.

User data collection

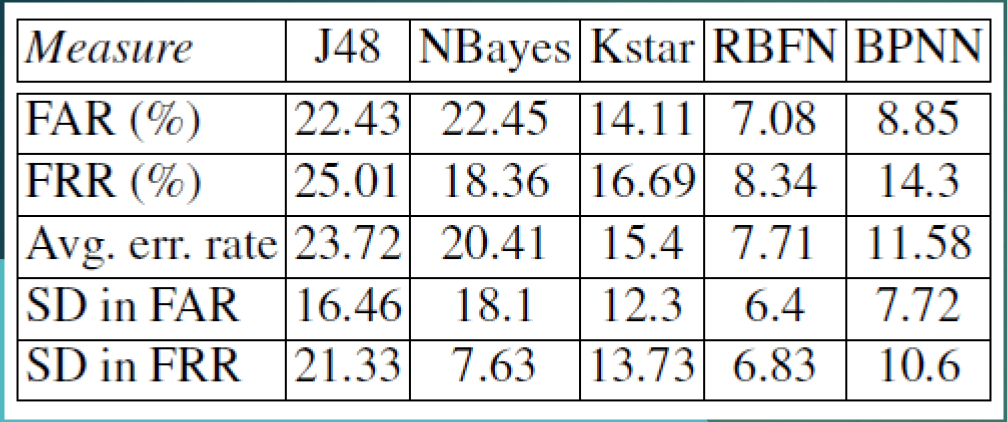
- *Methodology*. We had 20 Android phone users (12 female and 8 male) participate in our experiments and among the participants were students (85%) as well as professionals (15%).
- All participants were regular mobile phone users and ranged in age from 20 to 48 years.
- Before starting the collection, we described our objective to all participants and showed what kind of data would be collected. We asked participants to use the Android phones the same way they would use their own phones during the data collection period.
- Participants were asked to do the actual data collection outside of the lab, allowing them to get familiar with the phone first.
- Participants were asked to complete the collection of 6 sessions (with each session lasting 10 minutes) within 3 days, and they could use the phone freely as their own phones (e.g., using it to browse the web, install new software, etc.) during the entire collection period.

- Evaluation measures
  - False Acceptance Rate (FAR): indicates the probability that an impostor is classified as a legitimate user.
  - False Rejection Rate (FRR): indicates the probability that a legitimate user is classified as an impostor

- In practice, a trade-off is usually made between the false acceptance rate (security) and the false rejection rate (usability).
- In general, a false rejection is less costly than a false acceptance, since a higher false acceptance rate will lower the security level of the authentication system, while a higher false rejection rate will frustrate a legitimate user, which is still unfortunate but arguably less problematic than a lower security level.
- In terms of security and usability, both lower FAR and FRR are desirable

Evaluation Results

Table 2. Evaluation results for the tested classifiers.



Measure	J48	NBayes	Kstar	RBFN	BPNN
FAR (%)	22.43	22.45	14.11	7.08	8.85
FRR (%)	25.01	18.36	16.69	8.34	14.3
Avg. err. rate	23.72	20.41	15.4	7.71	11.58
SD in FAR	16.46	18.1	12.3	6.4	7.72
SD in FRR	21.33	7.63	13.73	6.83	10.6

1. The evaluation results show that for the data collected from our participants, the two neural network classifiers (RBFN and BPNN) have the best performance with an average error rate of 7.71% and 11.58%, respectively, compared to the other classifiers, which have average error rates of between 15% and 24%.

2. Although these experimental results are encouraging for the feasibility of our scheme, an average error rate of about 7.8% is still very high for real world systems. The reason for an error rate of around 7.8% is that the performance of the classifiers decreases as the variance of the feature datasets increases. Table 2 shows the standard deviation of the FAR and FRR for each classifiers, ranging from 7% to 22%.

- A more ideal classifier suitable for our system should therefore meet the following requirements:

(1) The classifier should provide a relatively small FAR and FRR (less than 5% each).

(2) The classifier should be economical in terms of computational power required, considering that it will be run on mobile devices with limited resources

(3) The classifier should be able to deal with the sometimes significant variations in the feature dataset

PSO-RBFN Classifier (1)

- To improve the performance of the classification when working on data with significant variations in a user’s behavior, we applied an algorithm that combines Particle Swarm Optimization (PSO) and an RBFN classifier.
- In our work, the RBFN classifier was selected for two reasons:

(1) RBFN has the lowest FAR and FRR compared to the other classifiers, as shown in Table 2;

(2) comparing the two neural network classifiers (RBFN and BPNN), RBFN has better accuracy and is faster when authenticating a user (e.g., fast in constructing models), which is a desirable property for applications that are run on resource-limited devices such as mobile phones.

PSO-RBFN Classifier (2)

- PSO was selected for the following two reasons:

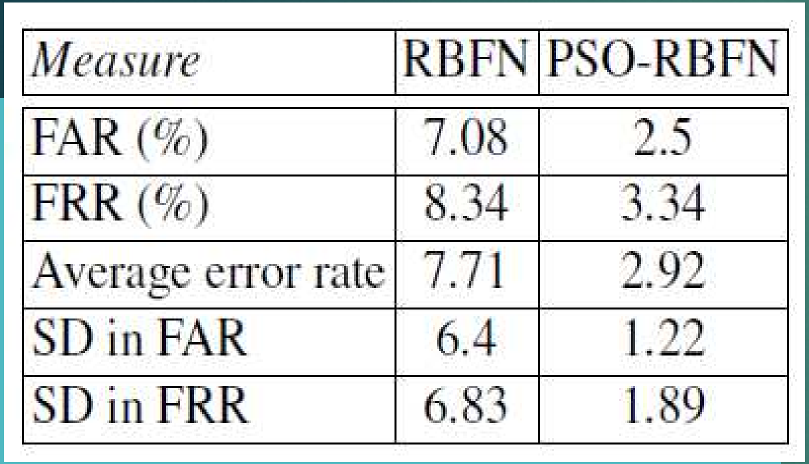
  (1) PSO is one of the most commonly used evolutionary algorithms used to optimize the structure of neural networks (e.g., RBFN) [51];

  (2) PSO can achieve faster convergence speed and requires fewer optimized parameters compared to other evolutionary algorithms such as Genetic algorithms, which benefits the implementation on a mobile phone. The principle of the PSO-RBFN classifier is described below.

- In hybrid PSO-RBFN, PSO can be used to enhance the RBFN training by optimizing the radial activation function and weighted sum of RBFN with a population-based iterative search procedure, so that PSO-RBFN can better deal with variations in a user's touch behavior compared to regular RBFN

PSO-RBFN Classifier (3)

Table 3. the experimental results of comparing the PSO-RBFN classifier against the regular RBFN classifier.



Measure	RBFN	PSO-RBFN
FAR (%)	7.08	2.5
FRR (%)	8.34	3.34
Average error rate	7.71	2.92
SD in FAR	6.4	1.22
SD in FRR	6.83	1.89

The numbers clearly show that using a combination of PSO and RBFN significantly improves the accuracy, reducing the average error rate from 7.71% for RBFN to 2.92% for PSORBFN.

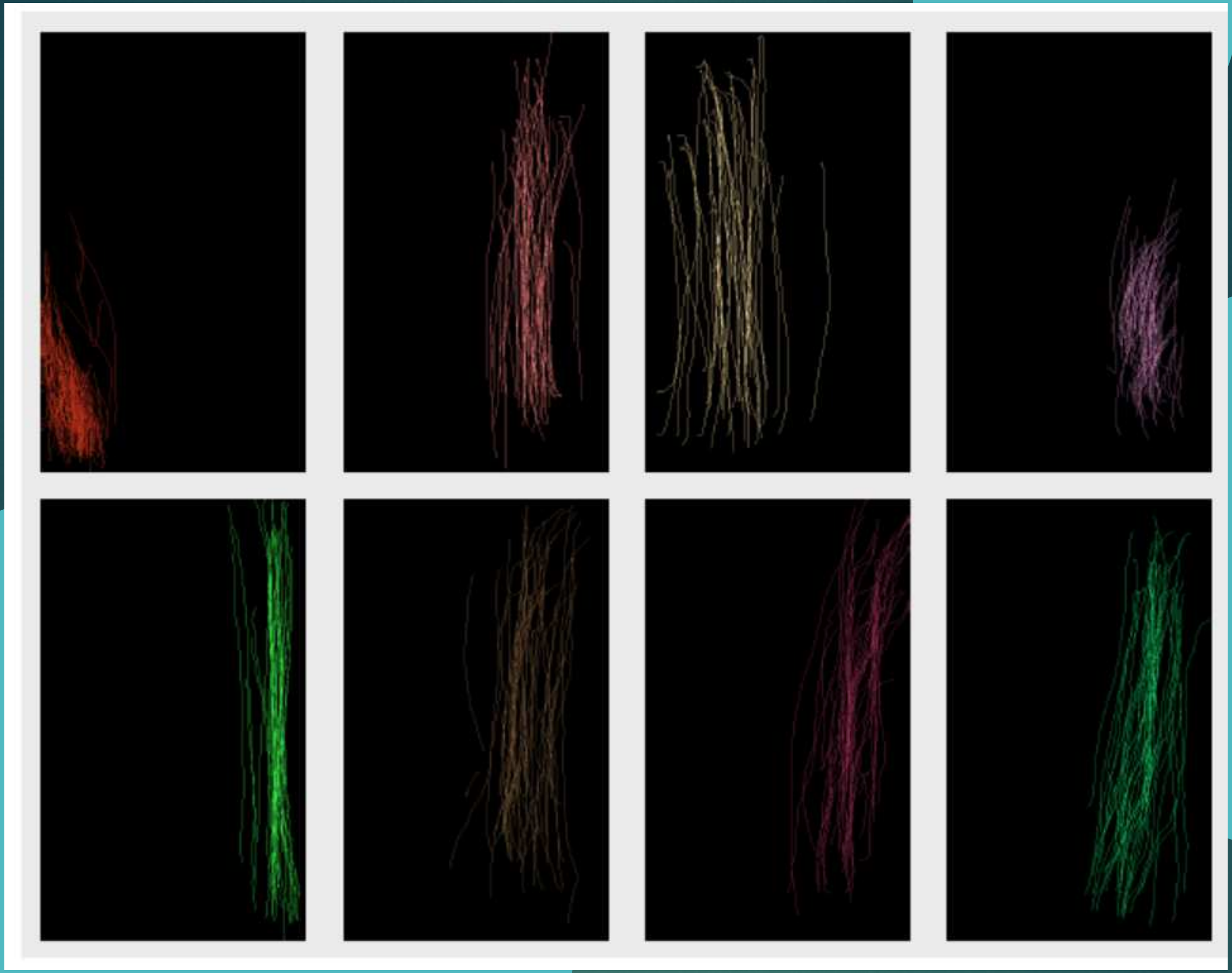
An FAR of 2.5% and FRR of 3.34% mean that the possibility of identifying an impostor as a legitimate user and the possibility of identifying a legitimate user as an impostor are low.

Furthermore, both the FAR and the FRR are below 5% when using the PSO-RBFN classifier and the standard deviation is also significantly lower compared to RBFN.

2012/2013

Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, Dawn Song, 'Touchalytics: On the Applicability of Touchscreen Input as a Behavioral Biometric for Continuous Authentication,` IEEE Transactions on Information Forensics and Security (Vol. 8, No. 1), pages 136-148, IEEE 2013.

- This work investigated whether a classifier can continuously authenticate users based on the way they interact with the touchscreen of a smart phone.
- Feasibility and Stability.



- Each user's interaction behavior on touchscreens can be quite unique. This figure depicts strokes recorded from eight different users, each reading three different texts on an Android phone. Geometric patterns that discriminate the users from each other are already apparent. Other differences might come from different stroke timing, pressure, and area covered on screen.

Enrollment Phase

- The main hypothesis of this study is that continuously recorded touch data from a touchscreen is distinctive enough to serve as a behavioral biometric.

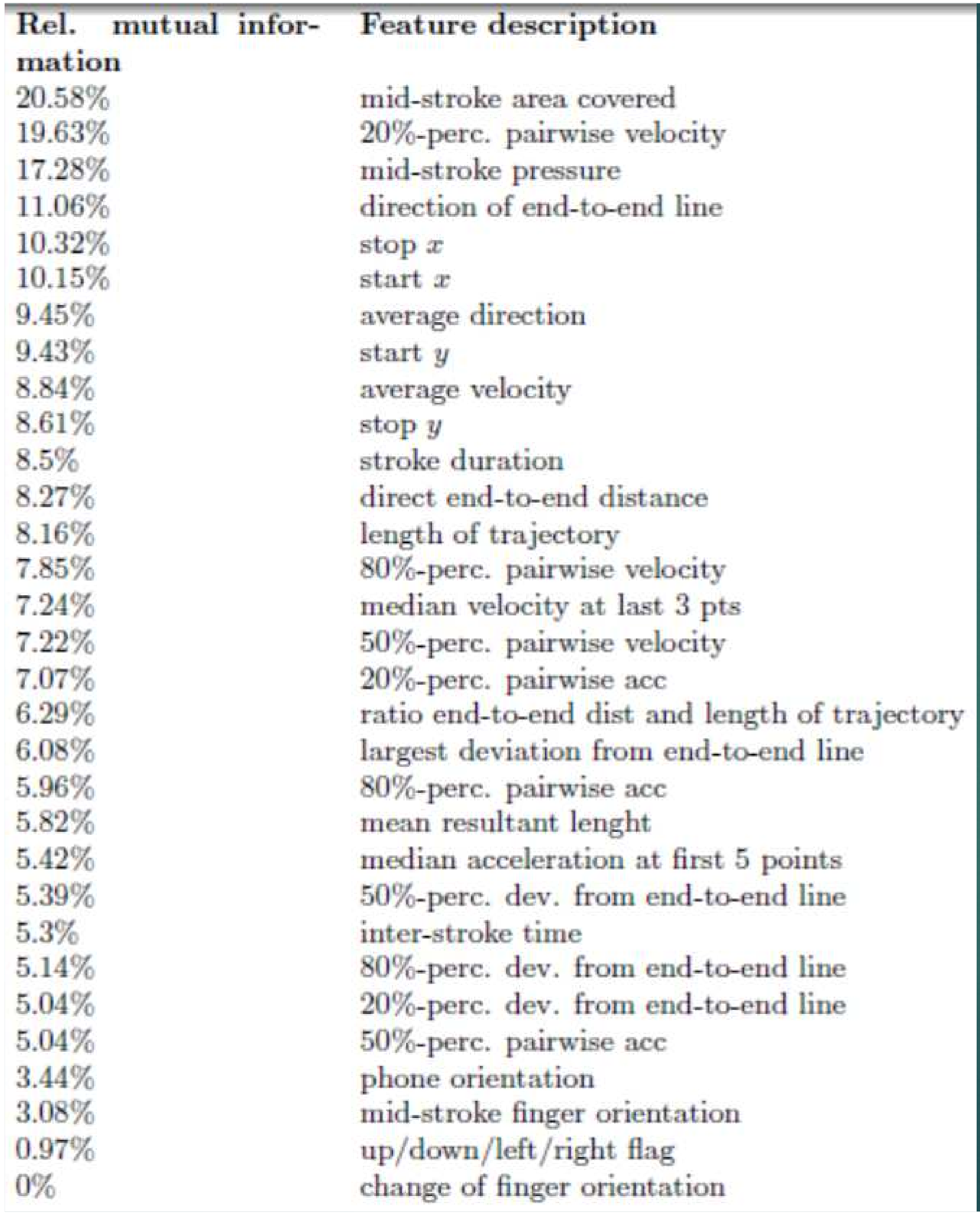
Define two particular user actions and call them `trigger-actions'.

- Sliding horizontally over the screen. Usually, one does this to browse through images or to navigate to the next page of icons in the main screen.
- Sliding vertically over the screen to move screen content up or down. This is typically done for reading email, documents or web-pages, or for browsing menus.

Continuous Authentication Phase

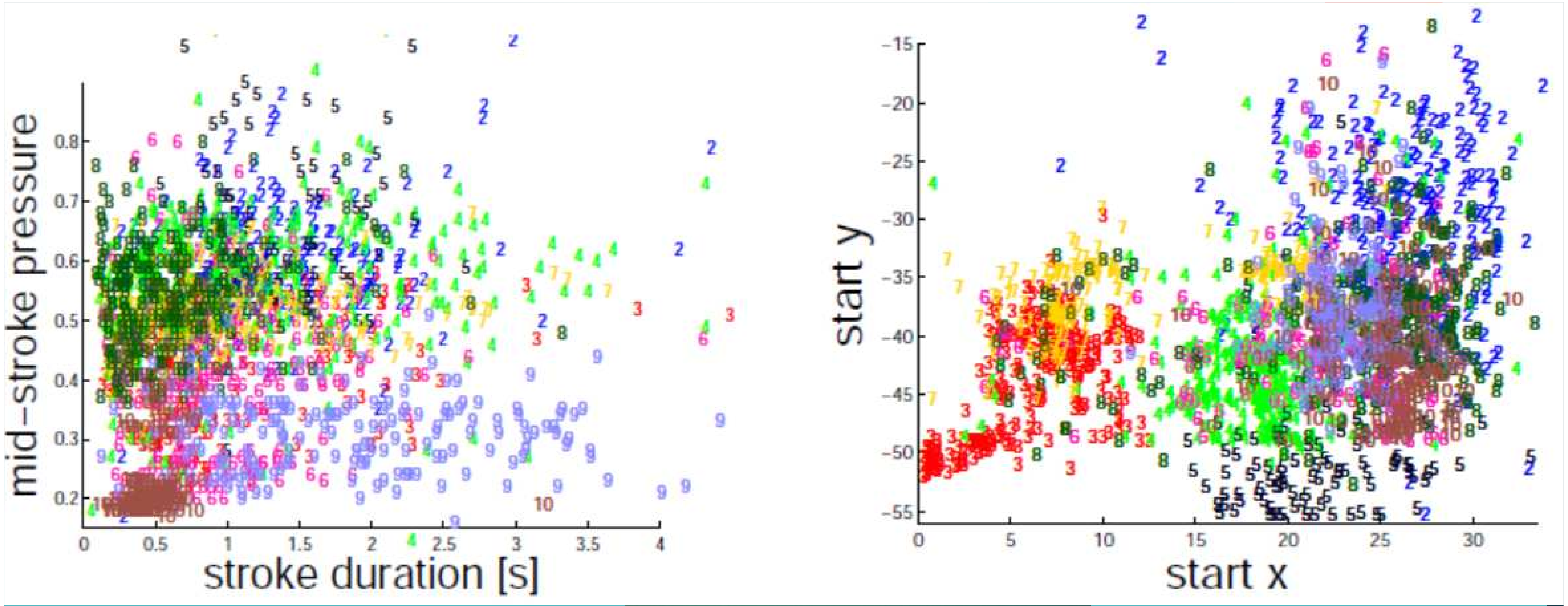
- Feature-extraction is to divide up the data records into individual strokes. A stroke is a sequence of touch data that begins with touching the screen and ends with lifting the finger.

- Once the classifiers are trained, the device begins the authentication phase. During this phase, the system continuously tracks all strokes and the classifier estimates if they were made by the legitimate user

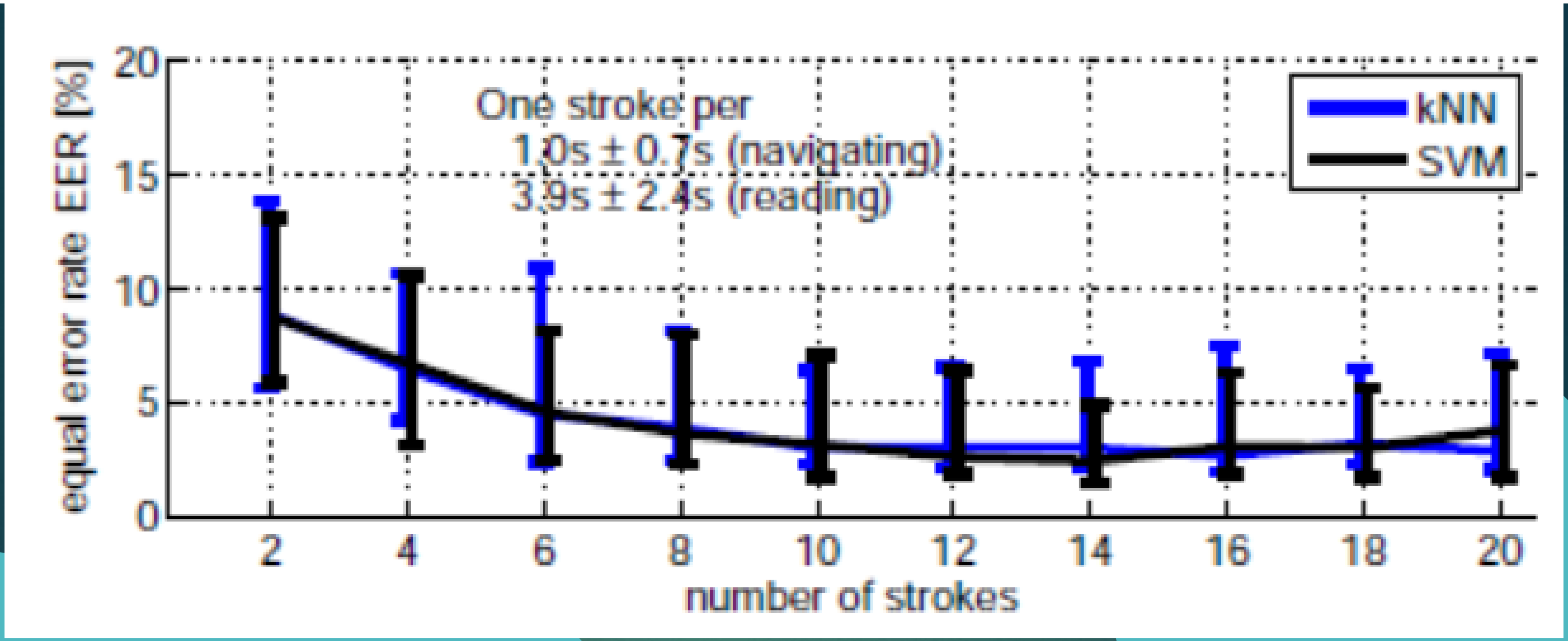


Rel. mutual information	Feature description
20.58%	mid-stroke area covered
19.63%	20%-perc. pairwise velocity
17.28%	mid-stroke pressure
11.06%	direction of end-to-end line
10.32%	stop x
10.15%	start x
9.45%	average direction
9.43%	start y
8.84%	average velocity
8.61%	stop y
8.5%	stroke duration
8.27%	direct end-to-end distance
8.16%	length of trajectory
7.85%	80%-perc. pairwise velocity
7.24%	median velocity at last 3 pts
7.22%	50%-perc. pairwise velocity
7.07%	20%-perc. pairwise acc
6.29%	ratio end-to-end dist and length of trajectory
6.08%	largest deviation from end-to-end line
5.96%	80%-perc. pairwise acc
5.82%	mean resultant lenght
5.42%	median acceleration at first 5 points
5.39%	50%-perc. dev. from end-to-end line
5.3%	inter-stroke time
5.14%	80%-perc. dev. from end-to-end line
5.04%	20%-perc. dev. from end-to-end line
5.04%	50%-perc. pairwise acc
3.44%	phone orientation
3.08%	mid-stroke finger orientation
0.97%	up/down/left/right flag
0%	change of finger orientation

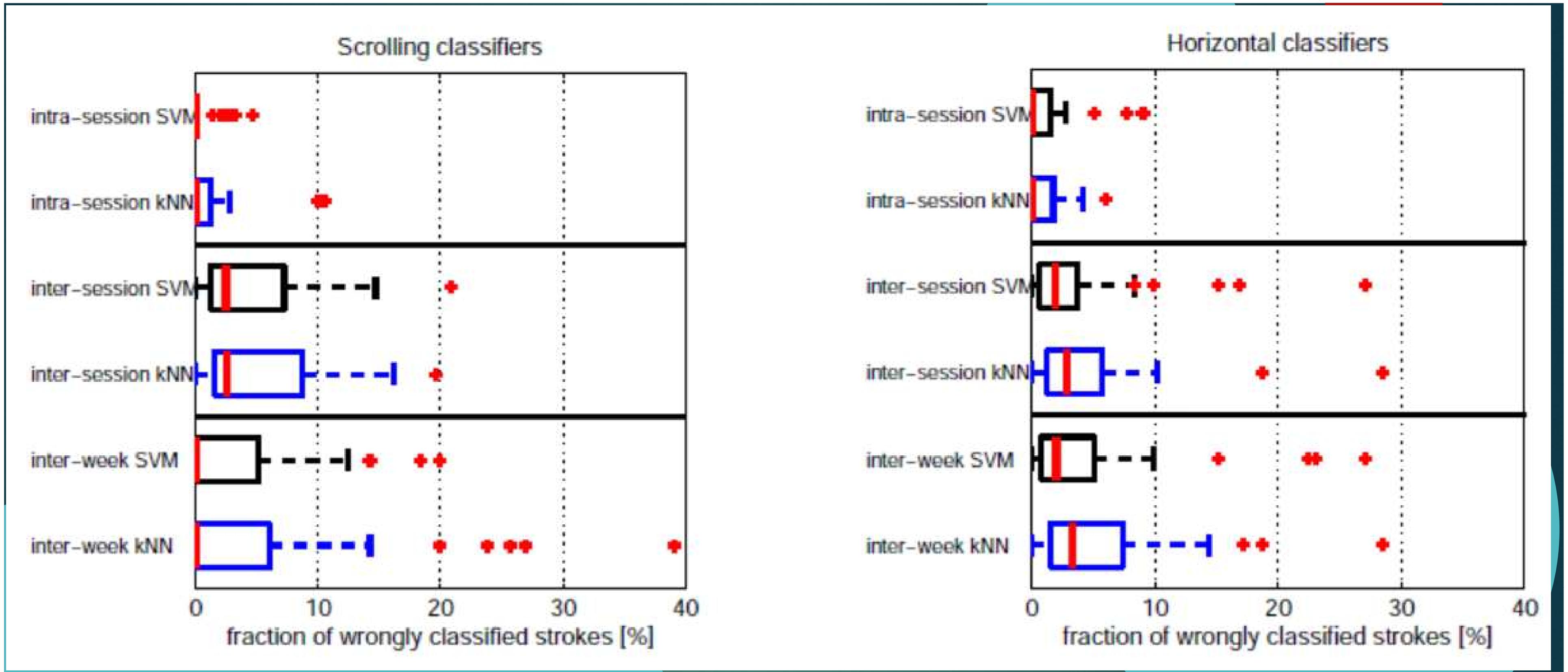
- A list of 30 features
- stroke velocity, fingertip pressure on screen and the direction of the stroke



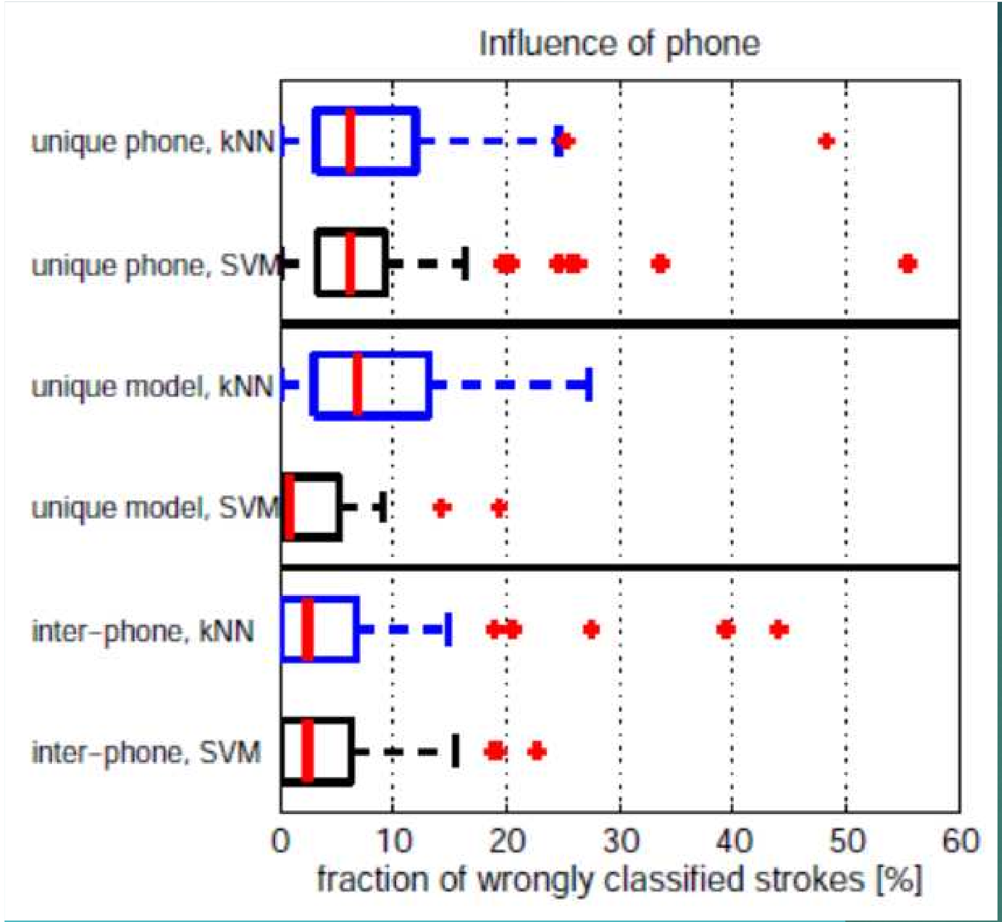
- Stroke features projected on a 2D-subspace. The user ID is given as a colored number. Already in these low-dimensional feature spaces, a class separation is apparent.
- The data depicted here was collected from users reading three Wikipedia articles in three different sessions. The left plot contrasts the finger pressure on the screen at the middle of the stroke against the stroke duration. The right plot shows the xy-positions where the fingertip first touches the screen.



- Two classifiers: Support vector machines (SVM) and k-nearest-neighbors (kNN)
- When deciding with a single stroke only, the EER is approximately 13%. Both classifiers obtain a lower error when increasing the number of strokes used to provide a classification output.
- At a level of 11 to 12 strokes, the EER converges to a range between 2% and 3% and stays there up to using 20 strokes.



- The median EER ranges from 0% to 4% across all usage scenarios. The median intrasession errors are 0%, whereas few outliers can reach a 10% EER.
- It seems that, within one session, most users do not considerably change their touch behavior.
- The inter-session EER reaches from 2% to 3% and the inter-week EER reaches from 0% to 4%, depending on the scenario and the classifier used.



The error rates for users on the same phone are on average 2% higher than for user data collected on multiple phones.

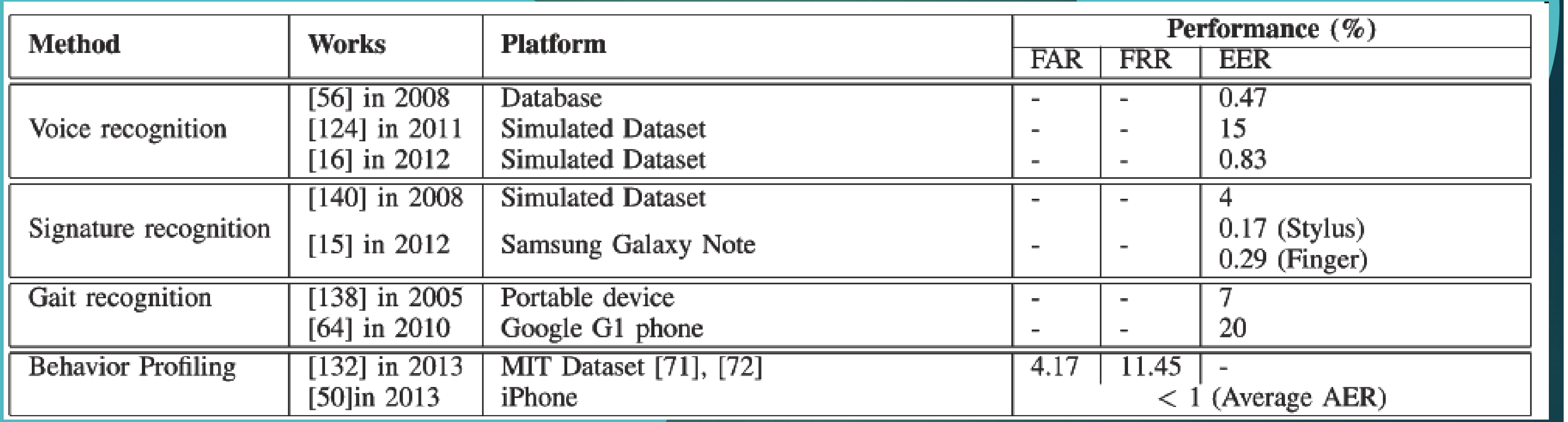
"While our experimental findings disqualify this method as a standalone authentication mechanism for long-term authentication, it could be implemented as a means to extend screen-lock time or as a part of a multi-modal biometric authentication system."

Outline

- Do We Need Biometric Authentication
- State-of-the-Art & Feasibility
- Reality and Behavioral Deviation
- How to Design a Robust Touch Behavioral Authentication

Behavioral Biometric Authentication (1)

Table shows the results reported by literature before 2015.

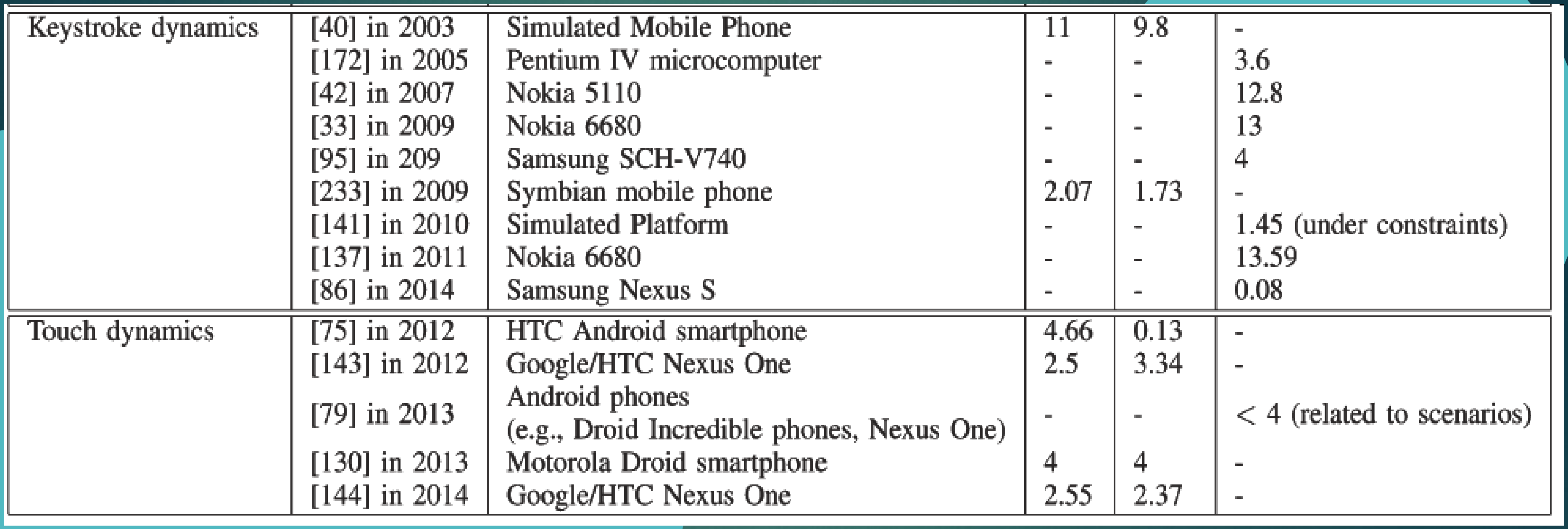


Method	Works	Platform	Performance (%)		
			FAR	FRR	EER
Voice recognition	[56] in 2008	Database	-	-	0.47
	[124] in 2011	Simulated Dataset	-	-	15
	[16] in 2012	Simulated Dataset	-	-	0.83
Signature recognition	[140] in 2008	Simulated Dataset	-	-	4
	[15] in 2012	Samsung Galaxy Note	-	-	0.17 (Stylus) 0.29 (Finger)
Gait recognition	[138] in 2005	Portable device	-	-	7
	[64] in 2010	Google G1 phone	-	-	20
Behavior Profiling	[132] in 2013	MIT Dataset [71], [72]	4.17	11.45	-
	[50]in 2013	iPhone	< 1 (Average AER)		

Weizhi Meng, Duncan S. Wong, Steven Furnell, and Jianying Zhou. Surveying the Development of Biometric User Authentication on Mobile Phones. IEEE Communications Surveys & Tutorials, vol. 17, no. 3, pp. 1268-1293, 2015.

Behavioral Biometric Authentication (2)

Table shows the results reported by literature before 2015.

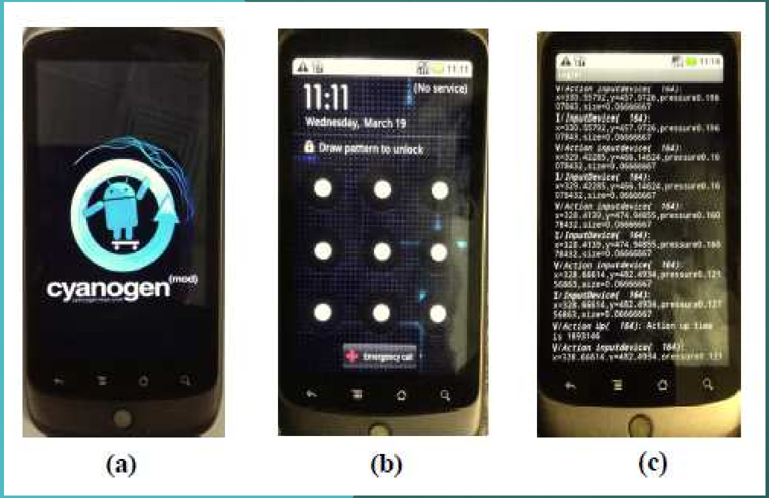


Keystroke dynamics	[40] in 2003	Simulated Mobile Phone	11	9.8	-
	[172] in 2005	Pentium IV microcomputer	-	-	3.6
	[42] in 2007	Nokia 5110	-	-	12.8
	[33] in 2009	Nokia 6680	-	-	13
	[95] in 209	Samsung SCH-V740	-	-	4
	[233] in 2009	Symbian mobile phone	2.07	1.73	-
	[141] in 2010	Simulated Platform	-	-	1.45 (under constraints)
	[137] in 2011	Nokia 6680	-	-	13.59
	[86] in 2014	Samsung Nexus S	-	-	0.08
Touch dynamics	[75] in 2012	HTC Android smartphone	4.66	0.13	-
	[143] in 2012	Google/HTC Nexus One	2.5	3.34	-
	[79] in 2013	Android phones (e.g., Droid Incredible phones, Nexus One)	-	-	< 4 (related to scenarios)
	[130] in 2013	Motorola Droid smartphone	4	4	-
	[144] in 2014	Google/HTC Nexus One	2.55	2.37	-

Study on Touch Movement (1)

A ***basic question*** here is how users would input patterns when performing touch movements on their phones.

- **Hypothesis 1.** Distinct users may perform the touch movement differently when inputting the patterns.
- **Hypothesis 2.** Through some input trials, one user's touch behavior may become more stable.



```
V/Action InputDevice( 184):
x=330.55792,y=457.9720,pressure0.196
07843,size=0.06666667
I/InputDevice( 184):
x=330.55792,y=457.9720,pressure0.196
07843,size=0.06666667
V/Action InputDevice( 184):
x=329.42285,y=466.14624,pressure0.16
078432,size=0.06666667
I/InputDevice( 184):
x=329.42285,y=466.14624,pressure0.16
078432,size=0.06666667
V/Action InputDevice( 184):
x=328.4130,y=474.94855,pressure0.160
78432,size=0.06666667
I/InputDevice( 184):
x=328.4130,y=474.94855,pressure0.160
78432,size=0.06666667
V/Action InputDevice( 184):
x=328.66614,y=482.4934,pressure0.121
56863,size=0.06666667
I/InputDevice( 184):
x=328.66614,y=482.4934,pressure0.121
56863,size=0.06666667
V/Action Up( 184): Action up time
is 1893146
V/Action InputDevice( 184):
x=328.66614,y=482.4934,pressure0.121
```

Figure. (a) The interface of CyanogenMod Android OS; (b) The screen of Android unlock patterns; (c) An instance of raw data collection.

Weizhi Meng, Wenjuan Li, Duncan S. Wong and Jianying Zhou. TMGuard: A Touch Movement-based Security Mechanism for Screen Unlock Patterns on Smartphones. The 14th International Conference on Applied Cryptography and Network Security (ACNS 2016), pp. 629-647, June 2016.

Study on Touch Movement (2)

Touch movement features:

- **The speed of touch movement (STM)**
- **The angle of touch movement (ATM)**

$$STM = \frac{\sqrt{(x2 - x1)^2 + (y2 - y1)^2}}{S2 - S1} \quad (1)$$

$$ATM \quad (d) = \arctan \frac{y2 - y1}{x2 - x1}, \theta \in [0, 360^{\circ}] \quad (2)$$

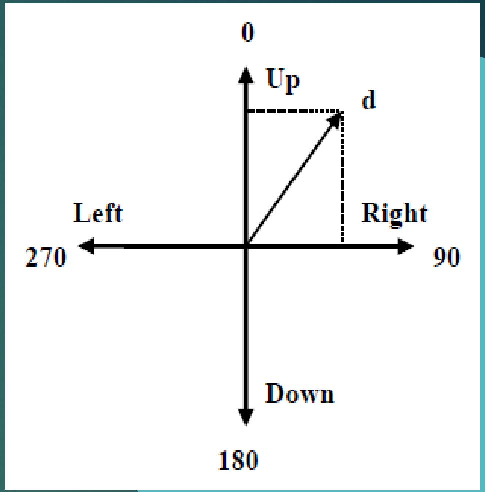
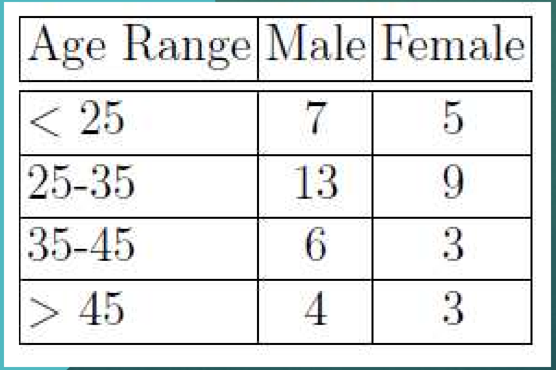


Figure. Directions for a touch movement.

Study Design and Results (1)

- **Phase1.** Each participant - 3 different patterns - re-enter three times (recorded) after two practice (not recorded) in one day.
- **Phase2.** We provide each participant with an Android phone equipped with our modified Android OS. Each participant should choose one of their created patterns in Phase1, and freely use the phone for another 2 days. After that, all participants were asked to return and input their patterns in our lab for three times.

Table 1. Participants information in the first user study (50 users)



Age Range	Male	Female
< 25	7	5
25-35	13	9
35-45	6	3
> 45	4	3

Study Design and Results (2)

- Users would perform differently when swiping their fingers on the touchscreen.
- Some users can perform stably, but not all!

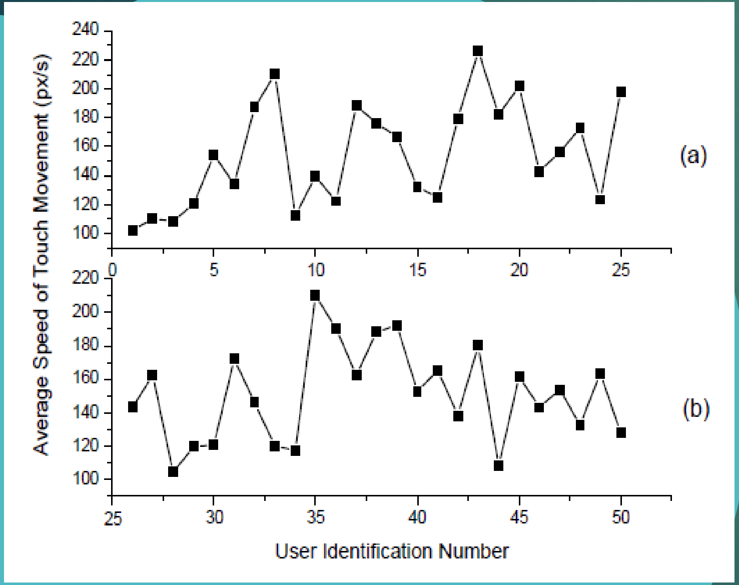


Figure. Average speed of touch movement (users from 1 to 50).

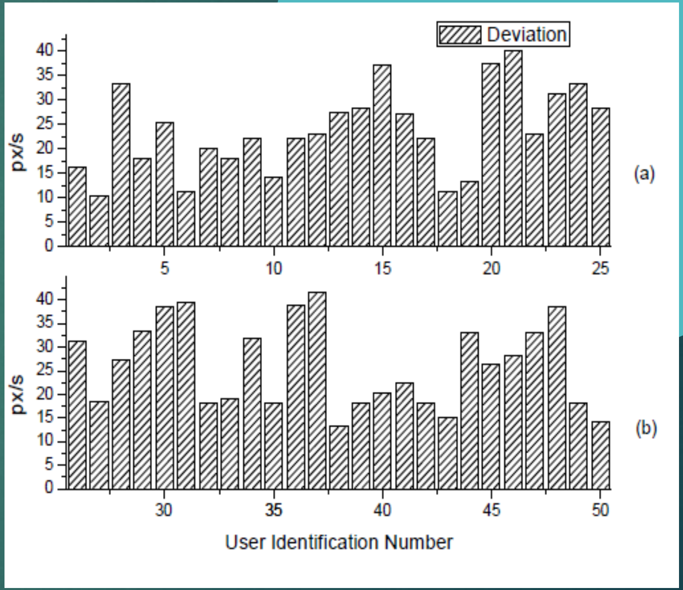


Figure. Deviation for average speed of touch movement (users from 1 to 50).

Study Design and Results (3)

- We further compute the deviations for all users when drawing the same pattern (3 trials for the same pattern).

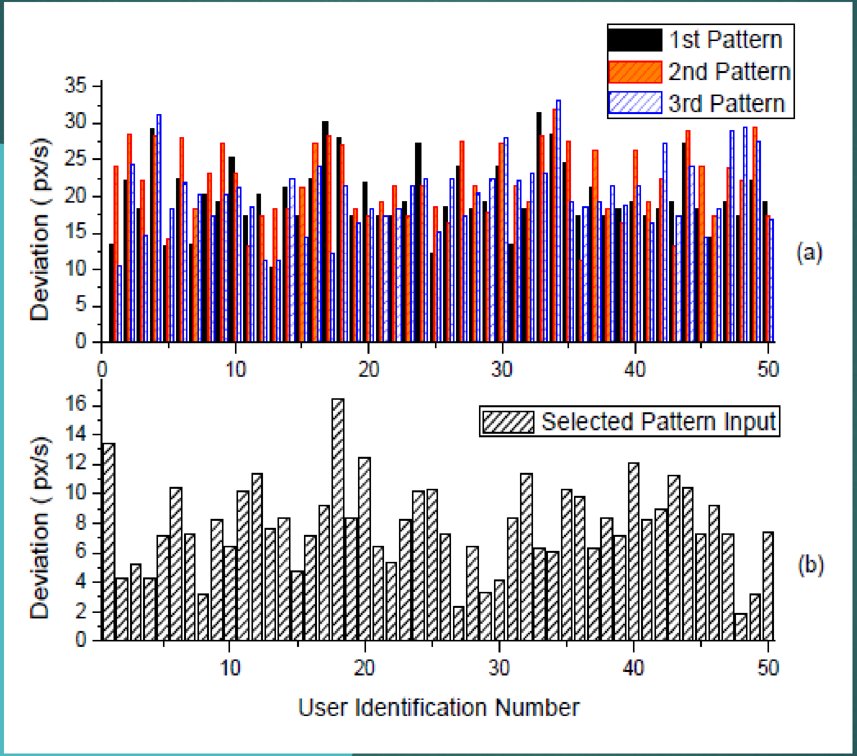
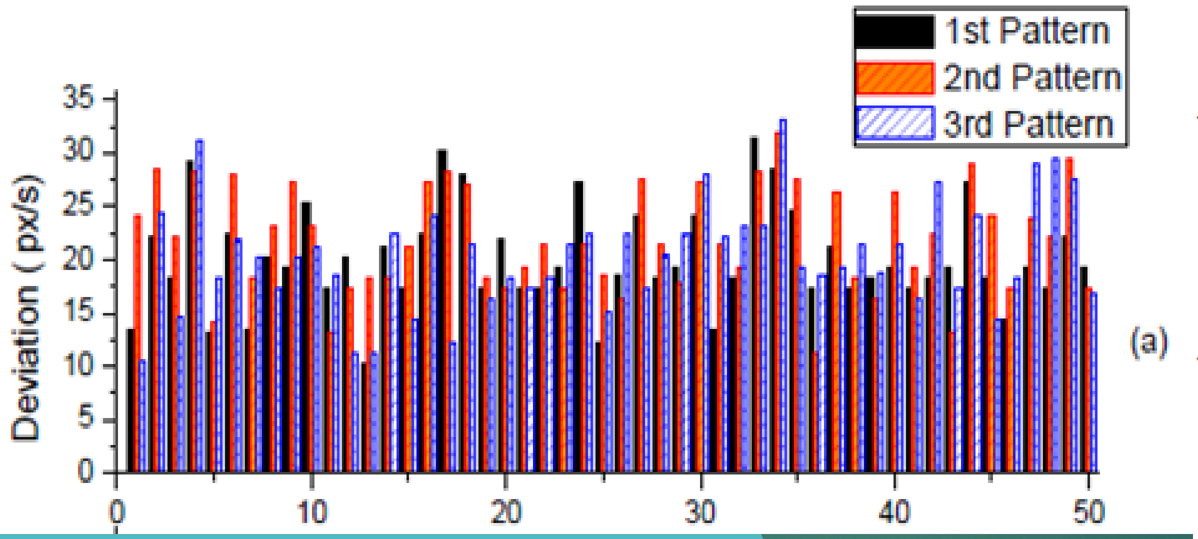
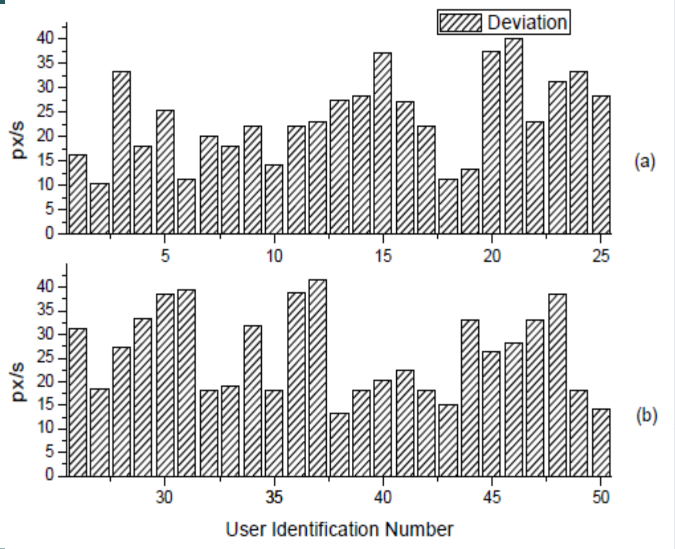


Figure. Deviation for average speed of touch movement (users from 1 to 50): (a) Deviation in Phase1 and (b) Deviation in Phase2.

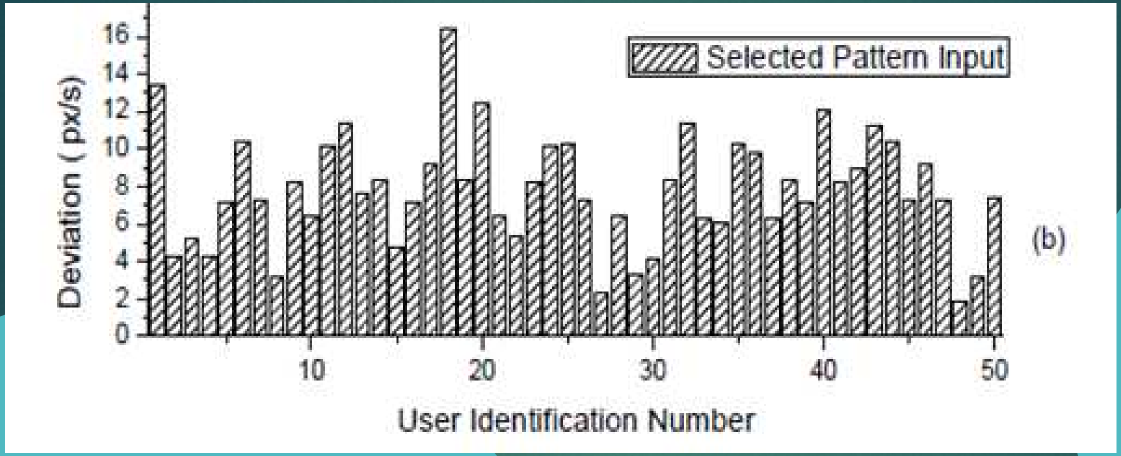
Study Design and Results (4)





- Deviations are lower than Figure 5.
- Users may perform different movement speeds according to distinct patterns.
- Nearly 75% deviations are below 25 px/s while only 3.3% deviations are over 30 px/s.

Study Design and Results (5)



- In Phase2, all users are required to input their selected patterns to unlock the phone for three times after a 2-day usage: at least 12 times.
- Only 6% deviations are over 12 px/s and up to 84% deviations are very close to, or even below 10 px/s.
- Users would perform a touch movement much more stably after a period of time.

TMGuard: A Security Mechanism for Android Unlock Patterns

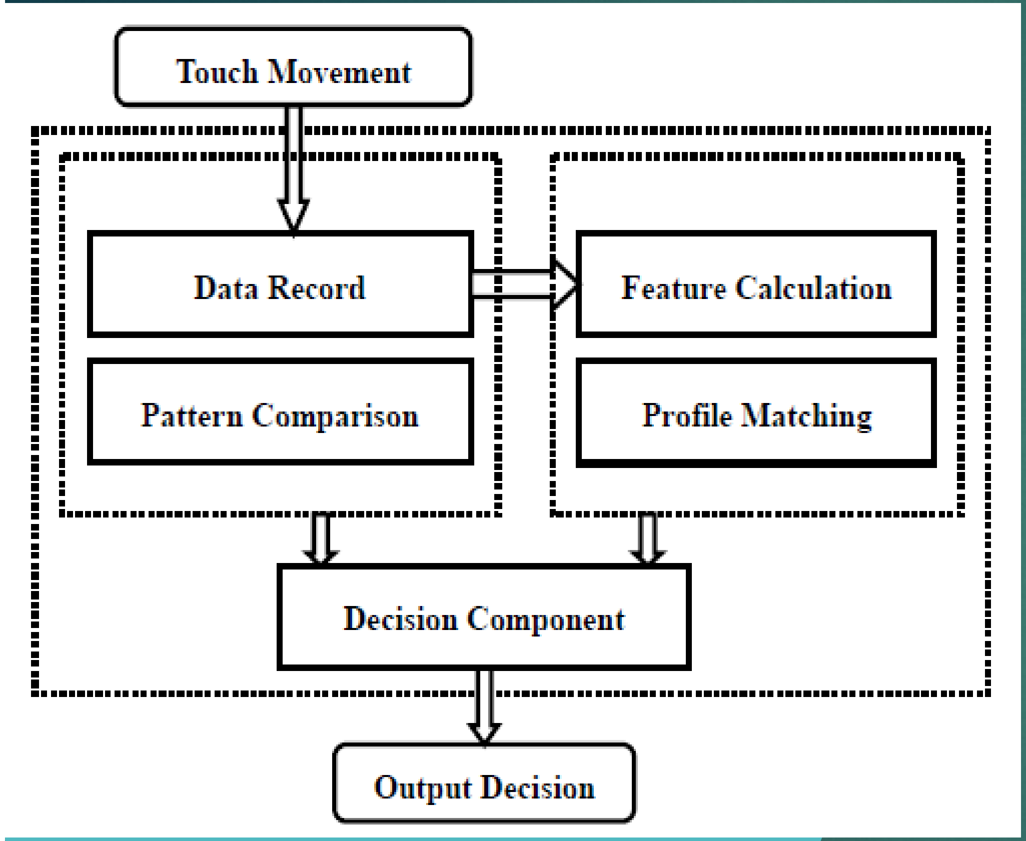
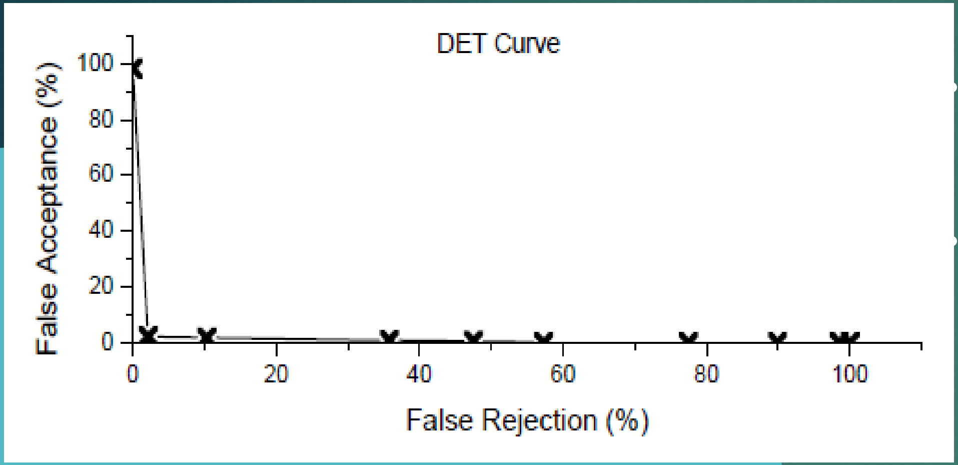


Figure. The high-level architecture.

- **Data Record.** To collect relevant data for speed and angle calculation.
- **Feature Calculation.** To calculate the speed and angle of a touch movement.
- **Pattern Comparison.** To compare the unlock pattern input with the stored pattern and report the result.
- **Profile Matching.** To build normal profile and make a comparison.
- **Decision Component.** To make the final decision whether the current user is legitimate.

Featured Result

Participants would pay attention to their touch behavior when inputting the patterns, but it is not a hard job.



- The FAR and FAR are computed by authenticating all users trials against their templates under different thresholds.
- It is seen that when the confidence threshold is 0.9, a FAR of 2.12% and FRR of 2.23% could be achieved.

Figure. Detection Error Tradeoff (DET) curve shows how FRR and FAR vary when different confidence thresholds are used.

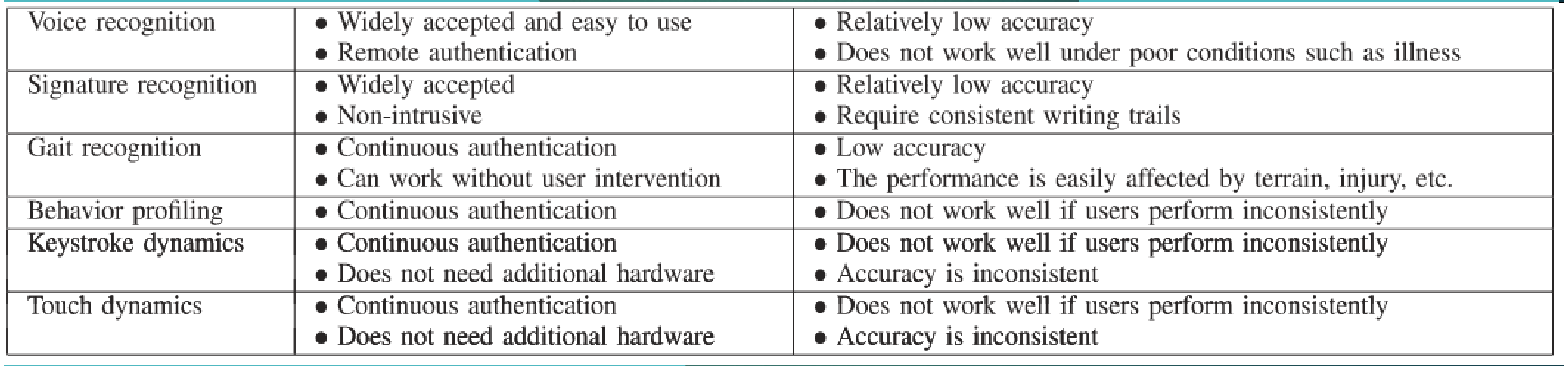
Summary

- It is feasible to apply behavioral biometrics to improving the security of Android unlock patterns.
- Users would perform a touch movement differently when inputting the patterns and they would perform more stably after inputting a pattern several times.
- It is noted that the average touch speed of some users may be similar.
- We believe that some parameters/features like the angle of touch movement (as a case study) can be combined to better distinguish users.

Outline

- Do We Need Biometric Authentication
- State-of-the-Art & Feasibility
- Reality and Behavioral Deviation
- How to Design a Robust Touch Behavioral Authentication

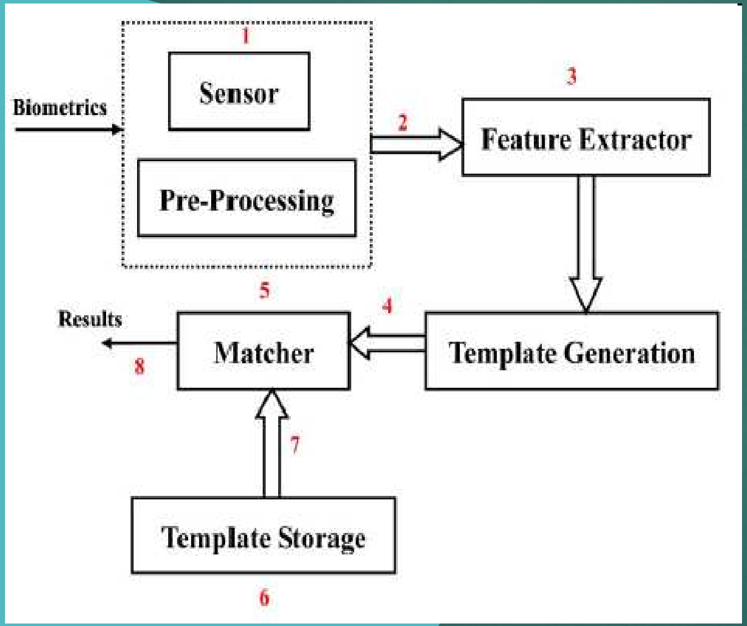
Advantages and Limitations



Voice recognition	• Widely accepted and easy to use • Remote authentication	• Relatively low accuracy • Does not work well under poor conditions such as illness
Signature recognition	• Widely accepted • Non-intrusive	• Relatively low accuracy • Require consistent writing trails
Gait recognition	• Continuous authentication • Can work without user intervention	• Low accuracy • The performance is easily affected by terrain, injury, etc.
Behavior profiling	• Continuous authentication	• Does not work well if users perform inconsistently
Keystroke dynamics	• Continuous authentication • Does not need additional hardware	• Does not work well if users perform inconsistently • Accuracy is inconsistent
Touch dynamics	• Continuous authentication • Does not need additional hardware	• Does not work well if users perform inconsistently • Accuracy is inconsistent

How to design a robust touch behavioral authentication

- Pay attention to the above vulnerable points, but system improvement is only one aspect!



- It is more important to guide phone users.

Open Challenges (1)

- Biometric authentication attempts to verify users according to either users' physical characteristics or behavioral habits.
- Although this kind of authentication has been developed over twenty years, there are still many challenges and open problems when authenticating users using biometrics.

- **Biometric feature selection**: To select an appropriate set of biometric features is a big challenge for biometric user authentication. Take touch dynamics as an example, many touch related features are available such as touch movement, touch direction, touch pressure, scroll, etc. In order to design a reliable authentication mechanism, how to select, decide and optimize an appropriate set of biometric features is a challenge and an open problem.

Open Challenges (2)

- **Algorithm development**: When having a set of biometric features, another challenge is how to develop an appropriate algorithm to improve or optimize the performance of authentication. Take behavior profiling as an example, the performance depends heavily on the designed algorithms that are used to generate pattern classification model. With the rapid development of computing, it is an important topic for designing more powerful algorithms for biometric authentication.
- **Users behavioral habit**: To authenticate users by means of the behavioral biometric authentication, a big challenge is that the authentication accuracy may be greatly decreased if the user performs very differently from his/her daily inputs (i.e., increasing false rates). This is a well-known and major limitation and an open problem for degrading the performance of behavioral biometric authentication.

Open Challenges (3)

- **Involved users**: To evaluate any biometric user authentication, involved users are a very important factor to affect the obtained results (i.e., different users may result in distinct patterns). Therefore, conducting a larger user study with even more users is always desirable. To enhance the evaluation, it is an important topic to explore how to conduct a systematic user study and experiment.

- **Evaluation platform**: A large number of biometric user authentication schemes have been proposed in literature aiming to improve the performance of authentication. However, it lacks of widely accepted and available benchmark in this area for comparing different works. To develop a standard evaluation platform in this area is a big challenge.

- **Leakage-resilient input**: Shoulder-surfing attacks are always a threat for user authentication, which use direct observation techniques such as looking over someone's shoulder, to get private information. Biometric authentication especially behavioral biometric authentication is vulnerable to such attacks, since an attacker can mimic users' behaviors by observation. Therefore, to design an appropriate method of leakage-resilient entry is very important.

Q&A

Thanks

weme@dtu. dk