Hold & Sign: A Novel Behavioral Biometrics for Smartphone User Authentication

Attaullah Buriro*, Bruno Crispo†*, Filippo DelFrari* and Konrad Wrona‡
*Department of Information Engineering and Computer Science (DISI),
University of Trento, Via Sommarive, 38123, Italy,
Email*:{attaullah.buriro, filippo.delfrari, bruno.crispo}@unitn.it
†DistrNet, KULeuven, Belgium
Email†: bruno.crispo@cs.kuleuven.be
‡NATO Communications and Information Agency, The Hague, Netherlands,
Email‡: konrad.wrona@ncia.nato.int

***Abstract*— The search for new authentication methods to replace passwords for modern mobile devices such as smartphones and tablets has attracted a substantial amount of research in recent years. As a result, several new behavioral biometric schemes have been proposed. Most of these schemes, however, are uni-modal. This paper presents a new, bi-modal behavioral biometric solution for user authentication. The proposed mechanism takes into account micro-movements of a phone and movements of the user's finger during writing or signing on the touchscreen. More specifically, it profiles a user based on how he holds the phone and based on the characteristics of the points being pressed on the touchscreen, and not the produced signature image. We have implemented and evaluated our scheme on commercially available smartphones. Using Multilayer Perceptron (MLP) 1-class verifier, we achieved $\approx 95\%$ True Acceptance Rate (TAR) with 3.1% False Acceptance Rate (FAR) on a dataset of 30 volunteers. Preliminary results on usability show a positive opinion about our system.**

***Index Terms*—Biometrics, Authentication, Human-Computer Interaction**

I. Introduction

Smartphones and tablets are widely used personal devices [1]. They generate and store an increasing amount of sensitive information. Furthermore, smartphones and tablets are used to perform security-critical transactions such as mobile payments, remote access to a company's intranet, etc. Existing authentication methods based on PINs and passwords are not convenient for the type of user interactions that characterize smartphones (very frequent [2] and short [3]); as a result, more and more users simply do not use any security (up to 40.9% according to a recent study [4]). Thus, the focus of security research has shifted towards biometric-based authentication schemes as a possible alternative. In particular, behavioral biometrics looks attractive since it is easy to implement because it requires only the standard hardware provided by most modern smartphones.

A handwritten signature establishes a user's identity based on how he writes his name. This behavioral modality is very popular because it is socially and legally accepted as a means of personal identification in everyday life, however its implementations require dedicated pads [5]. Modern touchscreens make it feasible to implement handwritten signatures in smartphones and tablets. However, like all other biometric modalities, this behavioral modality faces two basic challenges: intra-class variability and inter-class similarity. Intra-class variability refers to variations in signatures of the same person while inter-class similarity refers to the similarity of signatures of two or more persons found incidentally or intentionally as a result of an adversary's targeted attack. In smartphone user authentication scenarios, intra-class variation is challenging due to the comparatively smaller display area and the quality of the touchscreen, which result in large intra-class variations [6] [7]. Intra-class variations and inter-class similarity lead respectively to higher FRR and FAR.

This paper presents a smartphone user authentication system based on how a user holds his phone while signing on its touchscreen. The system profiles pressed screen points (so-called *touch-points*) and the micro-movements of the phone during the signing process in order to verify the user's identity. Although typing a PIN is easier than writing something on the touchscreen, a PIN can be forgotten, whereas most users remember their own name. Moreover, launching shoulder surfing and smudge attacks to steal PINs and passwords is relatively easy. In our method, even if an attacker knows what is being written, access is still denied because he cannot mimic the phone movements of the legitimate user.

We registered the phone micro-movements using multiple physical sensors available on most smartphones. These sensors are triggered when a user starts writing (first touch-point) and stop as the user finishes writing (last touch-point). We do not take into account the signature image because it can be copied and mimicked [8]. We tested our mechanism over a dataset collected from 30 users, by applying the anomaly detection (one-class) approach. Results show that using MLP as verifier, we achieve $\approx 95\%$ TAR and 3.1% FAR.

The main contributions of this paper are:

- The proposal and implementation of `Hold & Sign`, a new behavioral biometric user authentication mechanism, based on how the user holds his smartphone in his hand and signs his name on the smartphone touchscreen. It combines two behavioral modalities. Furthermore, it implements dynamic handwritten signature verification using multiple sensors that do not require the use of a

DOI 10.1109/SPW.2016.20

dedicated device to capture the signature.

- Experimental validation, considering how different situations in which a user can use the device can affect the robustness and accuracy of the biometrics.
- Performance and power consumption analysis during acquisition, training and testing phases. A preliminary usability analysis was carried out to assess how end-users reacted to our solution.

The rest of the paper is organized as follows. Section II describes related work. Section III presents our solution. Section IV illustrates the experimental analysis. Section V discusses the prototype implementation and some operational concerns such as power consumption. Section VI presents an analysis of users' experiences with the prototype. Section VII analyzes the results. Section VIII concludes the paper.

II. Related Work

Researchers have proposed several biometric-based solutions for smartphone user authentication. In this section, we survey the most relevant approaches.

A. *Sensor-Based Authentication*

Physical three-dimensional sensors – such as accelerometers, gyroscopes, and orientation sensors – are built into most smartphones. These sensors have been used to identify users based on their walking patterns [9], arm movements [10], arm movement and voiceprints [11], gesture models [12], and *free-text* typing patterns [13].

Li et al. [14] investigated the role of three sensors, namely accelerometer, orientation sensor, and compass, in addition to the touch gestures in continuous user authentication. They propose a transparent mechanism, which profiles finger movements and interprets the sensed data as different gestures. It then trains the Support Vector Machine (SVM) classifier with those gestures and performs authentication tasks. The authors achieved 95.78% gesture recognition accuracy on a database of 75 users.

Zhu et al. [12] propose a mobile framework *Sensec*, which makes use of sensory data from the accelerometer, orientation sensor, gyroscope, and magnetometer and constructs a user gesture model of phone usage. Based on this gesture model, *Sensec* continuously computes the sureness score, and authorizes the real users to enable/disable certain features to protect their privacy. Users were asked to follow a script, i.e. a sequence of actions; the sensory data was collected during the entire user interaction. *Sensec* identified a valid user with 75% accuracy and it detected an adversary with an accuracy of 71.3% (with 13.1% FAR) based on 20 recruited users.

Buriro et al. [13] authenticate users using a sensor-enhanced touch stroke mechanism based on two human behaviors: how a person holds his phone and how he types his 4-digit *free-text* PIN. Using a Bayesian classifier and a Random Forest (RF) classifier, they achieved 1% Equal Error Rate (EER).

A recent study [15] makes use of Hand Movement, Orientation, and Grasp (HMOG) to continuously authenticate smartphone users. HMOG transparently collects data from the accelerometer, gyroscope, and magnetometer when a user grasps, holds and taps on the smartphone screen. On a dataset of 100 test subjects (53 male and 47 female), HMOG achieved the lowest EER of 6.92% in *walking* state with the SVM verifier.

All the solutions given above use some of the three-dimensional sensors available in most smartphones and confirm the potential of these sensors for user authentication. Our solution uses 3-dimensional built-in sensors in combination with handwritten signatures to achieve high accuracy for authentication.

B. *Touch-Based Authentication*

User authentication based on touch-interaction is a comparatively less explored area. Touch-interactions can be used both for one-shot login and continuous user authentication [16]. Touch-based features may include time, position, the size of touch, pressure and touch velocity, etc. De Luca et al. [17] profile touch data generated during different slide operations for unlocking the smartphone screen. Using the Dynamic Time Warping (DTW) algorithm, they achieve 77% authentication accuracy.

Angulo et al. [18] suggest an improvement to the phone lock patterns. Their system authenticates users based on the lock patterns combined with the touch data associated with those lock patterns. They try multiple classifiers and they achieve an EER of 10.39% using a Random Forest classifier.

Sae-Bae et al. [19] use specific five-finger touch gestures. They achieve an accuracy of 90% on the Apple iPad. However, the method is not feasible for the small touchscreens of typical smartphones. Shahzad et al. [20] consider customized slide-based gestures to authenticate a smartphone's users. Their study yielded an EER of 0.5% with the combination of just three slide movements. Sun et al. [21] require users to draw an arbitrary pattern with their fingers in a specific region of the screen for unlocking their smartphones. Users were authenticated on the basis of geometric features extracted from their drawn curves along with their behavioral and physiological modalities. The solution in Sae-Bae and Memon [22] is conceptually similar to our work. This uni-modal online signature verification scheme extracts the histogram features from the user signature and performs user authentication. The lowest EER achieved was 5.34% across different sessions.

Our solution relies on the screen touch-points being pressed and the velocity of finger movement during the signing – neither signature image nor its geometry is used. It does not require the user to draw specific patterns for authentication, but simply use any pattern, which is convenient or well-known to him - e.g. to sign his name. This increases usability of our solution as the user is not required to perform an initial learning of an unknown pattern in order to memorize it and for his signing features to become stable and reliable.

C. *Signature-Based Authentication*

Some work has been done regarding signature-based biometric authentication on smartphones. Koreman and Morris

[23] propose a continuous authentication method based on multiple modalities, namely the face, voice, and signature on the touchscreen. Their study yielded an EER of 2.3%, 17%, 4.3% and 0.6% for voice, face, signature and fused modalities respectively.

Vahab et al. [24] implement online signature verification using an MLP classifier on a subset of Principal Component Analysis (PCA) features. The validation was performed using 4000 signature samples from the SIGMA database [25] and yielded an FAR of 7.4% and an FRR of 6.4%.

In recent work of Xu et al. [26], users were asked to write different alphabets on the screen; 42 handwriting features were extracted using a handwriting forensics approach (which focuses on the geometry of writing [27]). Those features were then classified using SVM. The proposed solution achieved an EER of 5.62%. Additionally, the touch slide (touch-points stimulated when writing an alphabet) yielded an EER of 0.75%.

Images of handwriting signatures have been used by SignEasy as an authentication method in iOS8 [28], allowing users to transparently add their electronic signatures on important documents. Similarly, a signature recognition system [29] performs user identification based on user signatures captured via a smartphone touchscreen or via a dedicated signature capturing device. It verifies signatures by computing the similarity score between the query signature and the stored signature template. Additionally, this system provides client-server solutions based on signature images. None of them uses phone movements and/or touch features for user authentication.

Our solution is different because it is bi-modal thus intuitively more secure than the uni-modal ones; it takes into account phone movements *and* finger movements during the signing process. Spoofing only one of the two modalities would not suffice to grant access to the phone.

III. Problem Formulation and Solution

In this section, we describe the main building blocks of our solution.

A. ***Threat Model***

We consider the adversarial model in which the attacker is already in possession of the device. The attacker can be a stranger who steals or finds the smartphone. Similarly, the attacker can be a family member, close friend or co-worker (who knows the implemented authentication mechanism). The goal of both types of attacker is the same: gaining access to the device and its contents. This threat model does not include the possibility of opening the phone and stealing a genuine biometric template. We do address this problem, by means of cryptography and trusted storage, however this issue is outside the scope of this paper.

B. ***Our Solution***

Our solution (see Figure 2) exploits the phone movements in hand and finger movements on the touchscreen as shown in Figure 1. In particular, we consider all the touch-points pushed for the entire signature and the velocity of the finger movement. All the physical sensors are triggered and kept running during the whole signing process (from first to last touch-point) on the touchscreen. Obtained sensor readings are then preprocessed to extract useful features. As we propose a bi-modal system, we need to combine the extracted features from both built-in sensors and the touchscreen to profile user behavior. Our model involves feature selection, which entails selecting the subset of productive features to be used for user authentication. A user profile template is formed based on the selected feature subset and is then stored in the main database. These behavioral vectors are later matched with the vector of the test sample in order to authenticate/reject the claimant.

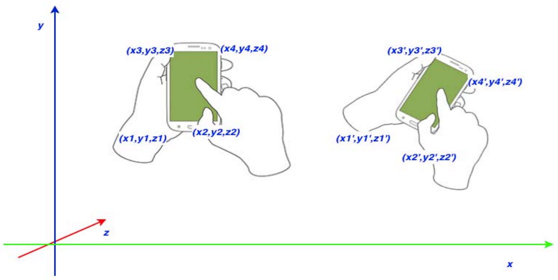


Fig. 1: Different phone positions during signing process

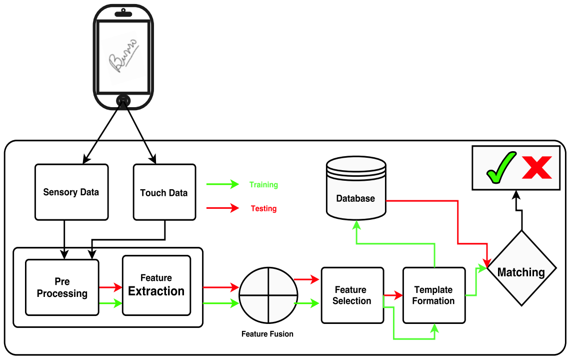


Fig. 2: Our proposed authentication system

C. *Considered Data Sources*

*1) **Sensors**:* Related work [13] [11] [30] [31] [15] shows that each user has a unique way of holding and/or picking up his smartphone. This movement behavior can be profiled only with three-dimensional sensors.

Our solution relies on three built-in three-dimensional sensors: the accelerometer, the gravity sensor, and the magnetometer. We derived two additional sensor readings from the accelerometer by applying two filters[1] (low pass and high pass) with the parameter $\alpha = 0.5$, and call the outcomes Low-Pass Filter (LPF) and High-Pass Filter (HPF) accelerometer readings. Thus, in total, we have three variants

[1] http://developer.android.com/guide/topics/sensors/sensors_motion.html

of accelerometer sensor readings: Raw, LPF, and HPF accelerometer readings. A raw accelerometer reading produces raw values including gravity values. An LPF accelerometer reading measures the apparent transient forces acting on the phone, caused by the user activity, and an HPF reading produces the exact acceleration applied by the user on the phone. The gravity sensor provides the magnitude and direction of the gravity force applied on the phone. The coordinate system and the unit of measurement of gravity sensor are the same as those of the accelerometer sensor. The magnetometer sensor measures the strength and/or direction of the magnetic field in three dimensions. It differs from the compass as it does not provide point north. The magnetometer measures the Earth's magnetic field if the device is placed in an environment absolutely free of magnetic interference. All the above sensors generate continuous streams in *x*, y and z directions. We have added a fourth dimension to all of these sensors and name it *magnitude*. Magnitude has been tested in the context of smartphone user authentication [13] [32] [15] and has proved to be very effective in classification accuracy. The magnitude is mathematically represented as:

$$S_M = \sqrt{(a_X^2 + a_Y^2 + a_Z^2)} \qquad (1)$$

where S_M is the resultant dimension and a_X, a_Y and a_Z are the accelerations along the X, Y and Z directions.

*2) **TouchScreen**:* The touchscreen provides the user interface for the operation of the device. Devices can be categorized as single or multi-touch devices. A finger and/or a pen interacts with the touchscreen. In Android, the library `MotionEvent` provides a class for tracking the motion of different pointers such as a finger, stylus, mouse, trackball, etc. The event triggered as a result of a touch is reported by an object of this class. This object may contain a specific action code such as the location of the touch in *XY* coordinates of the touchscreen, and information about pressure, size and orientation of the touched area. The action code represents the state of the touch action, e.g. `Action_Down` stands for the start of a touch action while `Action_Up` represents the end of a touch action. The Android `VelocityTracker` class is used to track the motion of the pointer on the touchscreen. The class methods, `getXVelocity()` and `getYVelocity()`, are used to acquire the velocities of the pointer on the touchscreen in the *X* and *Y* axes respectively.

*D. **Considered Classifiers***

Generally, the problem of user biometric authentication is solved in two ways: with binary classification (training with two classes) and anomaly detection (training with only the target class). Classifiers are very powerful in discriminating the true user from a given training set, whereas anomaly detectors check for deviation from the legitimate user's behavior and authenticate/reject on the basis of this deviation. In order to train a binary classifier, the biometric data from both the owner and the non-owner of the smartphone is required, which is an unrealistic assumption in the real world, since the sharing of biometric information between smartphone users may lead to privacy concerns. Hence, we used anomaly detectors (1-class verifiers) for user authentication [15] [33].

We chose four different verifiers, i.e. BayesNET, K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP) and Random Forest (RF), because they were found to be very effective in previous studies. BayesNET and RF verifiers were used with their default settings. However, the parameters of both MLP and KNN were optimized, because with default parameters they performed quite poorly. We used K = 3 in KNN and similarly used 3 hidden layers in MLP. We used all of our verifiers wrapped into Weka's metaclass classifier; the OneClassClassifier[2].

*E. **Success Metric***

True Acceptance Rate (TAR): The proportion of attempts of a legitimate user correctly accepted by the system.
False Acceptance Rate (FAR): The proportion of attempts of an adversary wrongly granted access to the system. It can be computed as $FAR = 1 - TRR$
False Rejection Rate (FRR): The proportion of attempts of a legitimate user wrongly rejected by the system. It can be computed as $FRR = 1 - TAR$.
True Rejection Rate (TRR): The proportion of attempts of an adversary correctly rejected by the system.
Failure to Acquire Rate (FTAR): The proportion of failed recognition attempts (due to system limitations). A reason for this failure could be the inability of the sensor to capture, insufficient sample size, number of features, etc.

IV. Experimental Analysis

*A. **Data collection***

Android supports data collection in both fixed and customized intervals after registering the sensors with *registerlistener()*[3]. Such intervals are often termed *Sensor_Delay_Modes*. There are four delays: *SENSOR_DELAY_FASTEST* with a fixed delay of 0s, *SENSOR_DELAY_GAME* with a fixed delay of 0.02s, *SENSOR_DELAY_UI* with a fixed delay of 0.06s and *SENSOR_DELAY_NORMAL* with a fixed delay of 0.2s.

We developed an Android application, `Hold & Sign`, which can be installed on any Android smartphone starting from version 4.0.4. We used *SENSOR_DELAY_GAME*, since we observed that *SENSOR_DELAY_NORMAL* and *SENSOR_DELAY_UI* were too slow and some of the sensors were not able to sense the user interactions in these two modes. *SENSOR_DELAY_FASTEST* mode could have been used as well, but it includes noise in the data collection.

We recruited 30 volunteers (22 male and 8 female); the majority of them are either Master's or Ph.D. students but not security experts. In order to have diversity, we recruited users from several nationalities. The purpose of the experiment and the description of our proposed solution was clearly

[2] http://weka.sourceforge.net/doc.packages/oneClassClassifier/weka/classifiers/meta/OneClassClassifier.html

[3] http://developer.android.com/reference/android/hardware/SensorManager.html

explained to each user individually. The process of data collection and how data are stored were carefully explained. Each volunteer provided explicit consent to participate in the experiment. We collected data in three different activities, *sitting*, *standing* and *walking* with *Google Nexus 5*.

B. Features

We gathered 4 data streams from every 3-dimensional sensor except touchscreen, and we extracted 4 statistical features, namely mean, standard deviation, skewness, and kurtosis, from every data stream. Data from every sensor was transformed into a 4 by 4 features matrix. In total we obtained 16 features from all four dimensions of each sensor. Similarly, we extracted 13 features from touchscreen data. The extracted features from touchscreen data are shown in Table I.

C. Features Fusion

In a study conducted by Jain et al. [34], the authors explain that there are five levels in a biometric system at which the acquired data can be fused: sensor; feature; match score; rank; and decision level. The fusion of data as early as possible may increase the recognition accuracy of the system. However, the fusion of data at sensor level may not yield better results because of the presence of noise during data acquisition. Thus fusion at feature level is expected to provide better results, because the feature representation communicates much more relevant information. The extracted feature set from the data from multiple sources can be combined to form a new feature set. We used fusion at the feature level, in order to provide the maximum amount of relevant information to our recognition system. The fusion of 16 features from each sensor makes a new feature vector and we call this feature vector the pattern of user's hold behavior. The length of this feature vector is 80 features (16 for each of the five used sensors). Similarly, the feature vector of sign behavior is small (13 features, extracted from the captured touch-points through the touchscreen) and we call it a sign pattern.

The length of the fused feature vector for both modalities becomes 93 features.

D. Feature Subset Selection

To avoid overfitting and address the curse of dimensionality issue, we performed feature subset selection. Feature subset selection is the process of choosing the best possible subset, i.e. the set that gives the maximum accuracy, from the original feature set. Note that even if we achieve the same accuracy with reduced features, smaller feature vectors decrease computation time and allow the classifier to decide faster.

We evaluated our feature set (93 features for fused behaviors) with Recursive Feature Elimination (RFE) feature subset selection methods. We relied on scikit-learn[4], a Python-based tool for data mining and analysis, for RFE feature subset selection.

TABLE I: List of selected features from touchscreen data

No.	Touch Features
1	StartX
2	EndX
3	StartY
4	EndY
5	AvgXVelocity
6	AvgYVelocity
7	MaxXVelocity
8	MaxYVelocity
9	STDX
10	STDY
11	DiffX
12	DiffY
13	EUDistance

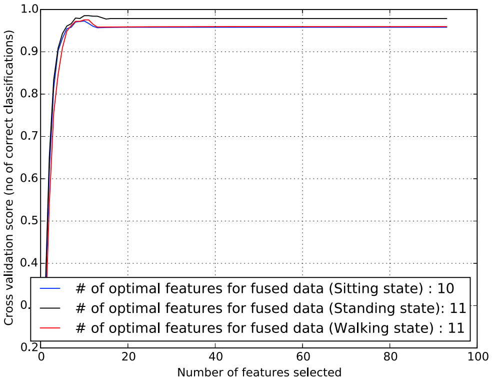


Fig. 3: RFE Feature Selection from *sitting*, *standing* and *walking* states.

The RFE classifier trains itself on the initial set of features and assigns weights to each of them. The features with smallest weights are later pruned from the current feature set. The procedure is repeated until the intended number of features is eventually reached[5]. We applied RFE with 10-fold stratified cross-validation using an SVM classifier on the data of all activities for two classes. The plot (see Figure 3) shows the optimal number (11) of features selected from fused data in standing and walking state and 10 for sitting state.

TABLE II: List of selected features from fused (bi-modal) data.

Sitting	*Standing*	*Walking*	*Combined*
MgX_Mean	HPFMag_Kurt	EndY	HPFY_Mean
RAWY_STD	RAWY_STD	RAWY_STD	HPFZ_Mean
DiffX	DiffX	DiffX	GrZ_Skew
StartY	StartX	RAWZ_Mean	StartX
MgY_Mean	EU_Distance	STDY	EndX
StartX	RAWMag_STD	HPFZ_Mean	StartY
EndY	StartY	StartY	EndY
MgMag_Mean	DiffY	HPFX_Skew	MaxYVelocity
GrY_Mean	HPFX_Mean	HPFX_Mean	AvgXVelocity
STDX	MgMag_Mean	DiffY	STDY
-	EndX	HPFY_Mean	DiffX

[4] http://scikit-learn.org/stable/

[5] http://scikit-learn.org/stable/modules/feature_selection.html

TABLE III: Results of different classifiers (averaged over all 30 users) in different activities.

	Sitting		*Standing*		*Walking*	
Classifiers	TAR	FAR	TAR	FAR	TAR	FAR
BN	0.758	0.001	0.740	0.003	0.710	0.000
MLP	0.797	0.001	0.790	0.004	0.790	0.000
IBk	0.761	0.001	0.750	0.002	0.720	0.000
RF	0.767	0.001	0.750	0.002	0.710	0.000

E. Analysis

We analyzed data in two settings, i.e. (i) a *verifying legitimate user* scenario, and (ii) an *attack* scenario.

In the *verifying legitimate user* scenario, we train the system with the data from the *owner* class and then test the system with the patterns belonging to that class. The outcome can be either accept or reject. We used a 10-fold stratified cross-validation method for testing. In cross-validation, the dataset is randomized and then split into k (here $k = 10$) folds of equal size. In each iteration, one fold is used for testing, and the other $k - 1$ folds are used for training the classifier. The test results are averaged over all folds, which give the cross-validation estimate of the accuracy. This method is useful in dealing with small datasets. Using cross-validation we tested each available sample in our dataset. We report the results of these settings in terms of TAR and FRR.

In the *attack* scenario, we train the system with all the data samples from the *owner* class and then test the system with the patterns belonging to all the remaining classes (29 users). The outcome can be either false accept or true reject. We report the results of these settings in terms of FAR and TRR.

F. Results

We report our results in three ways: intra-activity, inter-activity and activity fusion. By intra-activity, we mean training and testing each single activity (i.e. training walking to test walking only). Inter-activity means training with one single activity and using that training for testing all activities. We tested the training for each activity. In activity fusion, we used the combined data of all 3 activities for both training and testing (i.e. training with fused data from walking, sitting and standing) to test all activities. The reason for this is that we want to check whether training in a single activity is sufficient to recognize all the testing samples across activities. Otherwise, we would need to train the recognition system with patterns of multiple activities. As the MLP verifier has consistently out-performed all other verifiers in all three activities (see Table III), we will take into account only this verifier in further analysis.

The results of all settings are presented below:

*1) **Intra-Activity**:* The results of all three activities, prior to feature selection (averaged over 30 users), are given in Table III. We achieved $\geq 79\%$ TAR with full features in all the activities using the MLP verifier. We then applied a feature subset selection method (RFE) on our dataset. Figure 4 shows that we improved our authentication results (from $\geq 79\%$ to 85.56% in *sitting*, 86.75% in *standing* and 86% in *walking*) with our chosen RFE feature subsets (see Table II). We obtained 85.5% to 86.7% TAR with the MLP verifier in the three user activities. In related work, [15] reported 93.08% TAR but at the expense of 6.92% FAR using the 1-class SVM verifier and [11] reported 10.28% FAR and 3.93% FRR with 1-class RF verifier.

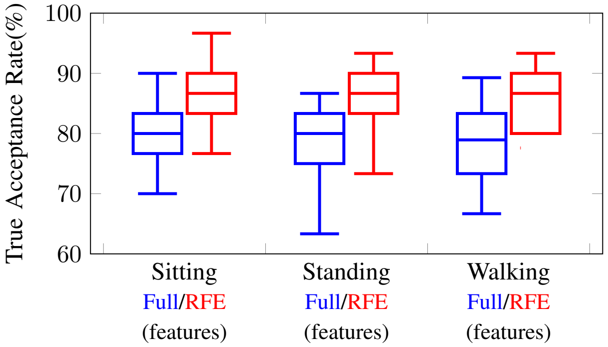


Fig. 4: Comparison of TAR for Full and RFE based feature subsets in *Intra-activity*.

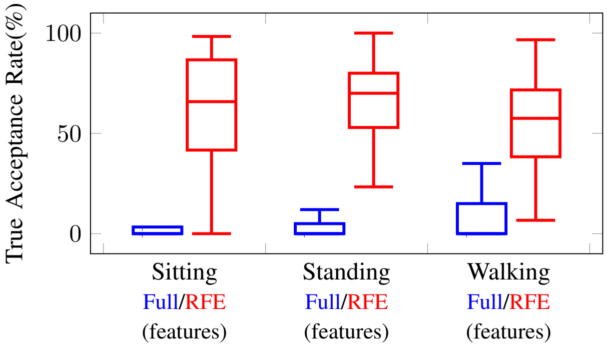


Fig. 5: Comparison of TAR for Full and RFE based feature subsets in *Inter-activity*.

*2) **Inter-Activity**:* In order to validate the applicability of our mechanism in multiple user positions, we tested its performance across multiple activities. For example, if we train the system with the training patterns of just the *sitting* activity and test it with the patterns of both *standing* and *walking* activities and vice versa, we can observe whether or not training with a single activity is sufficient. Figure 5 shows unsatisfactory results (65.82% at best), and thus we conclude that we need to train our system in multiple situations to increase its accuracy.

*3) **Activity Fusion**:* Training the system in just one activity and using it in multiple activities does not lead to good results. As a solution, we combined the patterns of multiple activities and applied the RFE feature selection method on the combined data. As done earlier, we picked 11 highly ranked features (see the last column of Table II) and proceeded to further analysis. We applied the same methodology (as per section IV-E) to test our combined dataset from all

TABLE IV: Results of MLP (averaged over all 30 users) for combined data of all three activities.

	Combined data from all activities			
Classifiers	TAR	FRR	FAR	TRR
MLP	0.948	0.052	0.031	0.969

three activities. The results are summarized in Table IV. The system achieved $\approx 95\%$ TAR at the expense of just 3.1% FAR. We observed that activity fusion could be useful in terms of usability (as it requires one-time training in multiple activities) and accuracy (we obtained $\approx 95\%$ TAR) so we checked its efficacy with the final implementation of *Hold & Sign*. We trained the system with a different set of training patterns from different activities and used the same set of features (see the last column of Table II) and compared the results.

V. Hold & Sign Implementation

We developed the final prototype of *Hold & Sign* taking into consideration all our findings. *Hold & Sign* uses the MLP classifier based on the feature set extracted using the RFE method. The analysis was performed using this application on a Google Nexus 5 smartphone running Android `4.4.4`. Screenshots for training and testing are shown in Figure 6. *Hold & Sign* requires a minimal configuration, i.e. a user may choose either both modalities or any one of them (as shown in Figure 6b) and needs to train the classifier accordingly. The user can also decide the number of training instances, i.e. how many times to write his own name on the touchscreen to train the classifier (Figure 6c). In all choices, the user is helped by the display of suggested recommended values. The user is later required to write his own name for authentication (see Figure 6d).

A. *Performance*

We tested the performance of *Hold & Sign*. We measured three different timings: sample acquisition time, training time and testing time. We computed these times for 3 different settings: with 15, 30 and 45 patterns. We tested each setting on the Google Nexus 5 with 35 tries for each time. Results are averaged over all 35 runs.

*1) **Sample Acquisition Time**:* This is the time used by the user to provide a sample for authentication. It is important to know it because users may feel annoyed by the required acquisition time that possibly results in complete removal of the *Hold & Sign* application. We compared the sample acquisition time for multiple mechanisms in Table V. What makes our acquisition fast is the free-text feature, e.g. the user can write any word (e.g. own name).

*2) **Training/Testing Time**:* Training time is the time required to train the classifier. It is usually computed just once, at the installation, when the training samples are provided to the system. In contrast, testing time is the time required by the system to accept/reject the authentication attempt. Our mechanism took 3.497s, 6.193s and 9.310s for classifier training with 15, 30 and 45 patterns, respectively. Similarly, the testing times with 15, 30 and 45 patterns were $0.200s$, $0.213s$, and $0.253s$, respectively. Comparison with the performance of other recent proposals is shown in Table VI.

TABLE V: Sample acquisition time for different methods adapted from [35].

Method	Sample Acquisition Time (s)
Our method	3.5
PIN	3.7
Password	7.46
Voice	5.15
Face	5.55
Gesture	8.10
Face + Voice	7.63
Gesture + Voice	9.91

B. *Power Consumption*

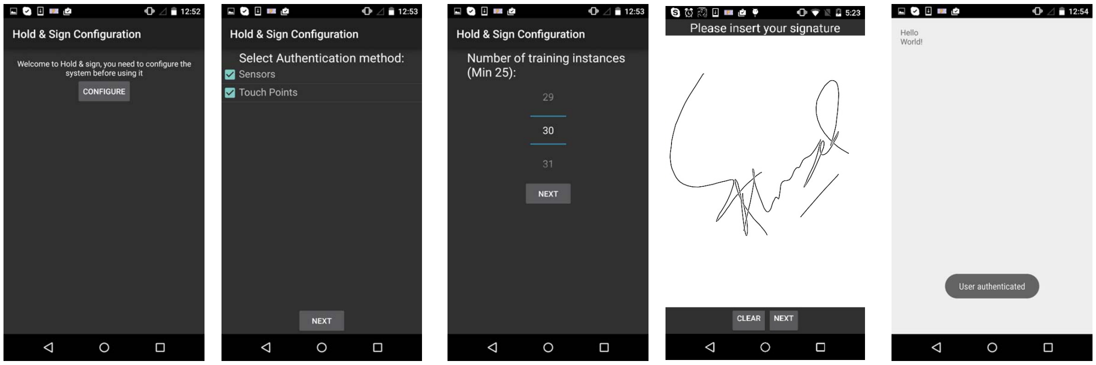
Generally, it is quite difficult to determine with high accuracy the power consumption of single mobile applications. Using dedicated hardware allows high accuracy [15]. However, there are software-based approaches that though less accurate, are being extensively used [36]. Since we wanted an initial indication, we used the software-based approach.

In order to check the overhead resulting from use of the application (in different steps), we terminated all the running applications and all Google services, switched off WiFi, Bluetooth, and cellular radios. The screen was kept running for the entire duration of the experiment with brightness at the lowest level and automatic brightness adjustment disabled. A similar approach is applied in [36]. We used *Trepn*[6] and performed the experiments as follows.

In the first step, we computed reference power consumption by running *Hold & Sign* with all the steps (sensor data collection, feature extraction, etc.) disabled. In the second stage, we enabled the sensor data collection part only to compute the overhead resulting from sensory data collection. In the third stage, we enabled the features extraction part to compute the power consumption resulting from this process. In the final step, we analyzed the app with all functionalities. We profiled the power consumption for all these settings of *Hold & Sign* for the entire duration (shortest duration 1 minute and 50s and longest 2 minutes and 40s) of the experiment with 35 attempts each. The reference power consumption is $460mW$. We observed 7.17% overhead ($493mW$) for sensor data collection, 27.8% in both data collection and feature extraction stages ($588mW$) and $\approx 1000mW$ in all stages of the final setting. The feature computation incurred just 19.2% overhead corresponding to data collection.

We observed that the average power consumption of our mechanism is very low, which makes it a power-friendly app. This claim can be supported by looking at some common smartphone tasks and their average power consumption [37] [38]:

[6] https://play.google.com/store/apps/details?id=com.quicinc.trepn&hl=en.


Hold & Sign Configuration

Welcome to Hold & sign, you need to configure the system before using it

Select Authentication method:

☑ Sensors

☑ Touch Points

Number of training instances (Min 25):

Please insert your signature

Hello World!

User authenticated

(a) (b) (c) (d) (e)

Fig. 6: Screenshots of *Hold & Sign* in training (a to d) and testing phase (d & e)

- A one-minute phone call: $1054mW$
- Sending a text message: $302mW$
- Sending or receiving an email over WiFi: $432mW$
- Sending or receiving an email over a mobile network: $610mW$

VI. Usability Analysis

We report the usability of our mechanism in two ways: based on how many patterns are enough for training the classifier to achieve significant authentication accuracy, and by applying standard the System Usability Scale (SUS) for collecting users' views about our proposed mechanism.

A. *Tradeoffs between Training and Accuracy*

As shown in Table V, the average duration of a signature drawn by a user on the touchscreen was 3.5s with the lowest value being 2s. In our test, we observed that the willingness of users to participate in our testing is strongly related to the amount of time spent for training. We expect a similar dependency also in normal usage. Hence it is important to evaluate the ratio of training time to accuracy. We observed that with just 15 patterns (in which case a user may take less than a minute to train the system), the user could be identified with around 70% TAR. Accuracy can be increased at the cost of training time. It took less than 4 minutes for the slowest of our testers to train the system with 45 patterns (15 in each activity) and authentication results were $\approx 90\%$. The TAR percents are averaged over 35 user attempts. The results are shown in Figure 7.

B. *Evaluation*

We distributed *Hold & Sign* along with an 11-question questionnaire adapted from the System Usability Scale[7] (SUS) to our chosen volunteers (30 users). The SUS assessment tool is widely used for gathering subjective impressions about the usability of a system. It has already been used in the context of smartphone authentication [35]. The response to each question can be given on a five-point scale ranging from 'Strongly Disagree' to 'Strongly Agree'. The SUS score is a value between 0 and 100 where a higher value indicates a more usable mechanism. A raw SUS score can be transformed to a percentile [40] or to a grading scale [41], allowing easier interpretation of results. The average SUS score is 68. Like the previous study [35], we added a question to this questionnaire: *What did you like or dislike about the mechanism?* This question was optional and subjective; users were to write some lines supporting the reason for like or dislike. We wanted to collect early feedback to allow us to improve our solution in future. We asked the users to use our app for some days (preferably a week) and share their experience with us. We received responses from 18 out of 30 volunteers (60%).

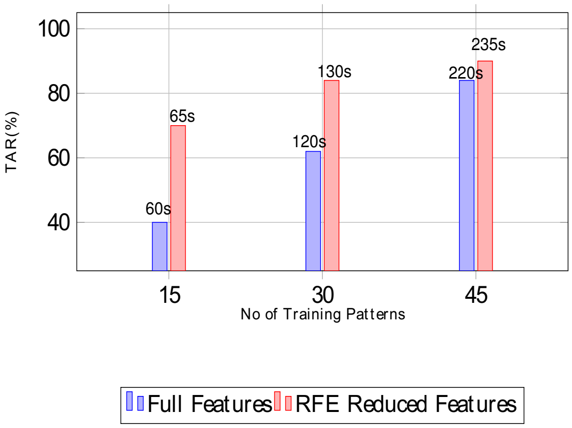


Fig. 7: User authentication on the prototype application. This figure verifies the average results obtained from the fusion of activities as described in Section IV-F3.The values above the bars indicate time spent to provide samples

C. *Responses*

We received useful feedback on our mechanism. We achieved an average SUS score of 68.33%. Our score is better than the well-established voice recognition score (66%) and its fusion with the face (46%) and gestures (50%) as reported

[7] http://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

TABLE VI: Comparison of our results with state of the art.

Ref.	Devices	Classifier	No. of Users	Training Time	Testing Time
Our method	Nexus 5	MLP	30	3.5 - 9.3s	0.215 - 0.250 s
Lee et al. [31]	Nexus 5	SVM	8	6.07s	20s
Li et al. [14]	Motorolla Droid	Sliding patterns	75	n.a	0.648s
Nickel et al. [39]	Motorolla Milestoon	KNN	36	90s	30s

in the literature [35]. Most of the responses were positive about the use of signing as an authentication credential. Most of the participants were also positive and comfortable using a finger and using the smartphone touchscreen (i.e. no complaints about the size of the display). We also got some negative responses, mostly related to the initial setup; it was "too cumbersome" for some, i.e. "a user has to sign multiple times in order to train the system whereas setting up a PIN is easier". We also received some negative responses regarding the system requiring the use of both hands.

Our mechanism is clearly in the initial stages and requires a lot of tuning. We are planning to incorporate these initial suggestions into future versions of *Hold & Sign* and also to run more extensive usability studies.

VII. Limitations

Our current solution suffers from two important limitations. Firstly, also pointed out by a volunteer, users must use both hands. One hand holds the phone and other hand's fingertip is used for the signature. The user, therefore, may experience some difficulty in using our solution, especially when on the move. Secondly, the system cannot predict the user's ongoing activity in order to extract the best pre-selected features and use them for verifying user identity.

VIII. Conclusion & Future Work

We proposed a new bi-modal behavioral biometric authentication, *Hold & Sign*, using as behaviors how a user holds a phone and how he writes on the touchscreen. We achieved 79% TAR at $zero$ FAR from 1-class MLP with full features in *walking* activity. The reason for this achievement could be the fact that during *walking*, sensors gather more data thus is possible to build accurate patterns. After applying feature subset selection, TAR improved to 86.7% at the expense of just 0.1% FAR. Lastly, processing the data from combined activities yielded 94.8% TAR at 3.1% FAR.

Hold & Sign requires on average just 3.5s to enter the behavioral pattern. Its ability to authenticate/reject a user within $0.215-0.250$s makes it very fast. The closest reported testing time in the literature is 0.648s [14].

Hold & Sign offers two advantages over traditional mechanisms. Firstly, a user can write his own name in an unconstrained way with a finger on the smartphone's touchscreen, which makes memorability and repetition easier.

There is no need to remember a password/pattern and no need to keep them secret, thus eliminating the problem of sharing and stolen passwords. Also, it is easy to integrate and implement in most modern smartphones without the need for additional hardware. *Hold & Sign* can be used as a standalone method or can be used in conjunction with other well-established mechanisms for additional security.

Since signature-based authentication is already deployed for user identification and it is also very common to use finger movements for navigating documents, e.g. web pages, photo albums, messages, etc., we expect our solution to receive positive user acceptance. The results of the preliminary usability analysis, with an SUS score above the average (68.33%), is a positive starting point.

As future work, we plan to investigate the permanency of this biometric modality, extend our work in terms of continuous authentication and explore its usability with a larger and more heterogeneous sample of testers. We are also going to address the problem of seamless and fast detection of a user's current activity since this would allow authenticating users based on the best feature subset selected from that particular activity.

Acknowledgment

The authors would like to thank all the participants of the experiment for their time and effort, colleagues for valuable and insightful input and anonymous reviewers for their reviews and comments.

This work was partially supported by the EIT Digital SecurePhone project and European Training Network for CyberSecurity (NeCS) grant number 675320.

References

[1] M. Böhmer, B. Hecht, J. Schöning, A. Krüger, and G. Bauer, "Falling asleep with angry birds, facebook and kindle: a large scale study on mobile application usage," in *proceedings of the 13th International Conference on Human Computer Interaction with mobile devices and services*. ACM, 2011, pp. 47–56.

[2] B. Spencer, "Mobile users can't leave their phone alone for six minutes and check it up to 150 times a day," *Daily Mail*, 11 Feb. 2013.

[3] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 179–194.

[4] M. Harbach, E. von Zezschwitz, A. Fichtner, A. De Luca, and M. Smith, "Itsa hard lock life: A field study of smartphone (un) locking behavior and risk perception," in *Symposium on Usable Privacy and Security (SOUPS 2014)*, 2014.

[5] M. M. Dıaz and U. Ingeniero de Telecomunicación, "Dynamic signature verification for portable devices," 2008.

[6] N. Houmani, A. Mayoue, S. Garcia-Salicetti, B. Dorizzi, M. I. Khalil, M. N. Moustafa, H. Abbas, D. Muramatsu, B. Yanikoglu, A. Kholmatov *et al.*, "Biosecure signature evaluation campaign (bsec'2009): Evaluating online signature algorithms depending on the quality of signatures," *Pattern Recognition*, vol. 45, no. 3, pp. 993–1003, 2012.

[7] M. Martinez-Diaz, J. Fierrez, J. Galbally, and J. Ortega-Garcia, "Towards mobile authentication using dynamic signature verification: useful features and performance evaluation," in *Proc. of the 19th Int. Conf. on Pattern Recognition*. IEEE, 2008, pp. 1–5.

[8] J. Galbally, *Vulnerabilities and attack protection in security systems based on biometric recognition*. Javier Galbally, 2009.

[9] J. Mäntyjärvi, M. Lindholm, E. Vildjiounaite, S.-M. Mäkelä, and H. Ailisto, "Identifying users of portable devices from gait pattern with accelerometers," in *proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, vol. 2. IEEE, 2005, pp. ii–973.

[10] 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," in *Proc. of the 6th ACM Symposium on Information, Computer and Communications Security*, 2011, pp. 249–259.

[11] A. Buriro, B. Crispo, F. Del Frari, J. Klardie, and K. Wrona, "Itsme: Multi-modal and unobtrusive behavioural user authentication for smartphones," in *proceedings of the 9th Conference on passwords (PASSWORDS 2015)*. Springer, 2016, pp. 45–61.

[12] J. Zhu, P. Wu, X. Wang, and J. Zhang, "Sensec: Mobile security through passive sensing," in *Int. Conf. on Computing, Networking and Communications (ICNC)*. IEEE, 2013, pp. 1128–1133.

[13] A. Buriro, B. Crispo, F. DelFrari, and K. Wrona, "Touchstroke: Smartphone user authentication based on touch-typing biometrics," in *New Trends in Image Analysis and Processing–ICIAP 2015 Workshops*. Springer, 2015, pp. 27–34.

[14] L. Li, X. Zhao, and G. Xue, "Unobservable re-authentication for smartphones." in *NDSS*, 2013.

[15] Z. Sitova, J. Sedenka, Q. Yang, G. Peng, G. Zhou, P. Gasti, and K. Balagani, "Hmog: A new biometric modality for continuous authentication of smartphone users," *arXiv preprint arXiv:1501.01199*, 2015.

[16] N. Sae-Bae, N. Memon, K. Isbister, and K. Ahmed, "Multitouch gesture-based authentication." *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 568–582, 2014.

[17] A. De Luca, A. Hang, F. Brudy, C. Lindner, and H. Hussmann, "Touch me once and i know it's you!: implicit authentication based on touch screen patterns," in *proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, pp. 987–996.

[18] J. Angulo and E. Wästlund, "Exploring touch-screen biometrics for user identification on smart phones," in *Privacy and Identity Management for Life*. Springer, 2012, pp. 130–143.

[19] N. Sae-Bae, K. Ahmed, K. Isbister, and N. Memon, "Biometric-rich gestures: a novel approach to authentication on multi-touch devices," in *proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, pp. 977–986.

[20] 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," in *proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 2013, pp. 39–50.

[21] J. Sun, R. Zhang, J. Zhang, and Y. Zhang, "Touchin: Sightless two-factor authentication on multi-touch mobile devices," *arXiv preprint arXiv:1402.1216*, 2014.

[22] N. Sae-Bae and N. Memon, "Online signature verification on mobile devices," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 6, pp. 933–947, 2014.

[23] J. Koreman, A. Morris, D. Wu, S. Jassim, H. Sellahewa, J. Ehlers, G. Chollet, G. Aversano, H. Bredin, S. Garcia-Salicetti *et al.*, "Multi-modal biometric authentication on the securephone pda," 2006.

[24] V. Iranmanesh, S. M. S. Ahmad, W. A. W. Adnan, S. Yussof, O. A. Arigbabu, and F. L. Malallah, "Online handwritten signature verification using neural network classifier based on principal component analysis," *The Scientific World Journal*, vol. 2014, 2014.

[25] S. M. S. Ahmad, A. Shakil, A. R. Ahmad, M. Agil, M. Balbed, and R. Anwar, "Sigma-a malaysian signatures database," in *IEEE/ACS International Conference on Computer Systems and Applications,(AICCSA)*. IEEE, 2008, pp. 919–920.

[26] H. Xu, Y. Zhou, and M. R. Lyu, "Towards continuous and passive authentication via touch biometrics: An experimental study on smartphones," in *Symposium On Usable Privacy and Security (SOUPS 2014)*. USENIX Association, 2014.

[27] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee, "Individuality of handwriting," *Journal of Forensic Sciences*, vol. 47, no. 4, 2002.

[28] Ananda. (2014) Signeasy announces new security features and enhancements for ios8 app, aims to streamline the way we access and sign digital paperwork. [Online]. Available: http://blog.getsigneasy.com

[29] Sutisoft. (2014) Signature verification. [Online]. Available: http://www.sutisoft.com/sutidsignature/key-features.htm

[30] W. Shi, J. Yang, Y. Jiang, F. Yang, and Y. Xiong, "Senguard: Passive user identification on smartphones using multiple sensors," in *IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2011, pp. 141–148.

[31] W.-H. Lee and R. B. Lee, "Multi-sensor authentication to improve smartphone security," in *International Conference on Information Systems Security and Privacy*, 2015.

[32] N. Zheng, K. Bai, H. Huang, and H. Wang, "You are how you touch: User verification on smartphones via tapping behaviors," in *International Conference on Network Protocols (ICNP)*. IEEE, 2014, pp. 221–232.

[33] D. Buschek, A. De Luca, and F. Alt, "Improving accuracy, applicability and usability of keystroke biometrics on mobile touchscreen devices," in *proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2015, pp. 1393–1402.

[34] A. K. Jain, A. A. Ross, and K. Nandakumar, *Introduction to biometrics*. Springer, 2011.

[35] S. Trewin, C. Swart, L. Koved, J. Martino, K. Singh, and S. Ben-David, "Biometric authentication on a mobile device: a study of user effort, error and task disruption," in *proceedings of the 28th Annual Computer Security Applications Conference*. ACM, 2012, pp. 159–168.

[36] H. Khan, A. Atwater, and U. Hengartner, "Itus: an implicit authentication framework for android," in *proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 507–518.

[37] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone." in *USENIX annual technical conference*, vol. 14. Boston, MA, 2010.

[38] W. Lee. (2013) Mobile apps and power consumption basics. [Online]. Available: https://developer.qualcomm.com/blog/mobile-apps-and-power-consumption-basics-part-1

[39] C. Nickel, T. Wirtl, and C. Busch, "Authentication of smartphone users based on the way they walk using k-nn algorithm," in *8th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*. IEEE, 2012, pp. 16–20.

[40] J. Sauro. (2011) Measuring usability with the system usability scale (sus). [Online]. Available: http://www.measuringu.com/sus.php

[41] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008.