Effect of Training Data Size on Touch Keystroke Verification with Medians Vector Proximity Classifier

Shatha J. Alghamdi
Computer Science Department
Faculty of Computing & Information Technology
King Abdulaziz University, Jeddah, Saudi Arabia
sjalghamdi@kau.edu.sa

Lamiaa A. Elrefaei[1,2]
[1]Computer Science Department
Faculty of Computing & Information Technology
King Abdulaziz University, Jeddah, Saudi Arabia
[2]Electrical Engineering Department
Faculty of Engineering, Shoubra
Benha University, Cairo, Egypt
lamia.alrefaei@feng.bu.edu.eg

***Abstract* - This paper presents a user verification system on mobile phones that is based on keystroke dynamics derived from a touchable keyboard. The touch keystroke dynamics dataset are collected using a developed mobile application in which, unlike other systems, no specific text is required. Two scenarios were considered: few-training and more-training datasets. The Median Vector Proximity classifier is applied on both datasets and the performance of the system is investigated using a different number of features. Using few-training dataset, the average EER were 12.9% and 12.2% for 31 and 33 features respectively. Using more-training dataset brings improved results with EER=0.76% and EER=0.39% for 31 and 33 features respectively. The Medians Vector Proximity becomes more accurate when increasing the training data. Also, using more features reduced the average EER by 0.7% and 0.37% in few-training and more-training datasets respectively. The proposed system is compared against other systems and shows promising results.**

Keywords - Touch Keystroke Dynamic; Biometric; Mobile Authentication; Soft Keyboard; Touch Gesture; Touch Screen.

I. INTRODUCTION

The mobile phone is one of the essential things for people. These devices are not only used for calling or sending text messages, they are also used in many applications such as accessing the internet (social networking, e-banking or e-commerce), sending and receiving emails and storing sensitive documents [2]. Many of these applications require the user to establish their identities on the phone. On a mobile device, the user authentication must act quickly, easily and user-friendly.

Biometrics meets these authentication criteria, so they become more preferable. Most of behavioral biometrics are collected implicitly and are cheaper than the biological ones [3]. Keystroke dynamics is a behavioral biometric that checks what you type and how you type it [4].

The literature has explored the feasibility of using keystroke dynamics and typing pattern behavior for user authentication for personal computers (PCs) [5]. Unlike PCs and old mobile phones, the touch screen is the primary input medium on the latest mobile phones and tablets [5]. However, the finger gesture on the touch screen has additional features which could be detected through a touch screen.

Al-Jarrah [6] proposed a personal computer anomaly detector for keystroke dynamics (hardware keyboard). His proposed system was based on a statistical measure of proximity depending on the median and standard deviation.

Part of this work was presented at CICSyN2015 [1]

The author evaluated his proposed system (Median Vector Proximity) using the CMU dataset [7]. In this paper, we propose applying Median Vector Proximity [6] on touch keystroke data derived from mobile's touch screen (touchable keyboard). In this case, we built our own dataset since there is no available touch gesture dataset [3, 4, 7, 8]. We collected data based on two techniques (will be explained in section III.B). The main contributions of this work are:

- Using mobile's touch screen to detect touch keystroke dynamics.
- Building our own touch gesture dataset by using an Android application.
- Don't specify what should the participants write during the experiment (dynamic text).
- Investigating the performance of the median vector proximity classifier using different numbers of touch gesture feature.
- Evaluate the proposed system and compare it with the previous work.

The rest of this paper is organized as follows: the related work is introduced in section II. Our proposed system is explained in section III. The system evaluation and results are discussed in section IV. Finally, a conclusion and a future work are shown in section V.

II. RELATED WORK

The literature showed that the keystroke dynamics and typing pattern behavior - detected from personal computer users - are feasible for user authentication [5]. Based on a proximity statistical measures, **Al-Jarrah** [6] proposed an anomaly detector for keystroke dynamics. The author evaluated his proposed system (Median Vector Proximity) using the CMU dataset which used the password (***.tie5Roanl***) for their experiments [7]. Using 31 features detected from hardware keyboard (personal computers), it has shown a lower error rates (EER=8%) compared with other 14 detectors in [7].

In the other hand, several studies found that the features extracted from touch-screen could be used efficiently to differentiate between different users [3, 7, 8]. In [8], **Julio Angulo and Erik Wästlund** built an android application to collect the data about how users draw specific lock patterns on a touch-screen. Two main features were used: finger-in-dot time, which is the time interval from the moment the user's finger touches a dot until the finger is moving outside the dot area. The other feature was finger-in-between-dots time, which is the finger's speed when moving from one dot to another. By analyzing data from 32 users using different classifiers (Euclidian, Manhatten, Mahalanobis, Recursive Partitioning, Supportive Vector Machine and Random Forest), they found that a Random Forest classifier attained the best result by giving an average EER of 10.39%. This percentage approved that finger movements to draw the lock pattern on touch-screen could be used for identification purposes.

There are few types of research that focus on touch keystroke features. **Nan Zeng et al.** [4] proposed a non-intrusive user verification mechanism using a 12-key touchable keyboard. The acceleration, pressure, touch size, key-hold and inter-key time feature sets are collected from 80 users through an Android application. Their approach was based on removing the outlier information so a small amount of raw data is filtered out. The authors proved the uniqueness of each user pattern. They used a nearest neighbor algorithm to classify data. The EER for a 4-digit password is about 3.65% while it is between 4.55% and 4.45% for an 8-digit password. The authors also tested how the four sets of features contribute to the final accuracy, and they found that the combination of all feature sets always outperforms individual feature set.

Xuan Huang et al. [10] proposed a mobile authentication system which is based on specific username and password (username="abertaytest" password="abertay2011") but also combines the typing behavior recognition. The proposed system detects the keystroke latency and key hold-time features from a touchable QWERTY-keyboard. To classify these features, authors used a statistical approach. This system has four alert levels (low, medium, high and very high). By testing the system on 40 users with different alert levels, they found that EER = 7.5%.

Margit Antal et al. [11] found that the classification and verification accuracy was improved by adding touch screen based features. They tested their system on 42 users using two types of android mobiles: Nexus7 tablet and Mobile LG Optimus L7 II P710 device. Each user types a specific password (***.tie5Roanl***), they extracted 41 purely touch keystroke features and 71 features (mixed of keystroke and touch features). For identification, they used Naïve Bayes, Bayesian Networks, C4.5(J48), k-NN, SVM, Random Forest and MLP classifiers. The random forest showed 82.5% and 93% of accuracy by using 41 and 71 features respectively. For Verification, they used Euclidean, Manhattan and Mahalanobis classifiers. The Manhattan provided 15.3% and 12.9% of EER by using 41 and 71 features respectively.

All the work in [3, 4, 7-10] is done on a touchable keyboard and they specified a specific word or numbers to be entered by the participants. In this paper, we propose a user verification system on a mobile phone using the touch keystroke data detected from the touch screen (touchable QWERTY-keyboard) as in [10] and applying Median Vector Proximity classifier [6] on it.

By exploring the literature [3, 4, 7, 8], there is no available touch gesture dataset. So, we developed a mobile application for collecting the required data and built our own dataset.

III. PROPOSED WORK

In this section, the proposed system structure is presented first, then the data collection techniques are explained, and then the computed features is discussed and finally the details of the user verification using median vector classifier is presented.

A. Proposed system

We propose a mobile user verification system which depends on mobile's touch screen for detecting keystroke dynamics. Like any biometric system, this system consists of some modules [12] as shown in Fig. 1.

The system has two phases: Enrolment and Authentication. Here the sensor that will be used for collecting data is the mobile's touch screen. The extracted features are introduced in section III.C. In the Matching module, the median vector classifier [6] will be used and its details are discussed in section III.D.

B. Data Collection

In order to collect touch data, we built an android application –using Android Development Tools (ADT) with Eclipse- which runs as a stand-alone application on android mobile. This application collects the touch data from a touchable keyboard.

We conducted the experiment on only one mobile device to make sure that the data is consistent. Here we used an HTC one M8 Android phone with a Super LCD3 capacitive touch screen, and 1440 x 2560 pixels (5.5 inches) to perform the experiment.

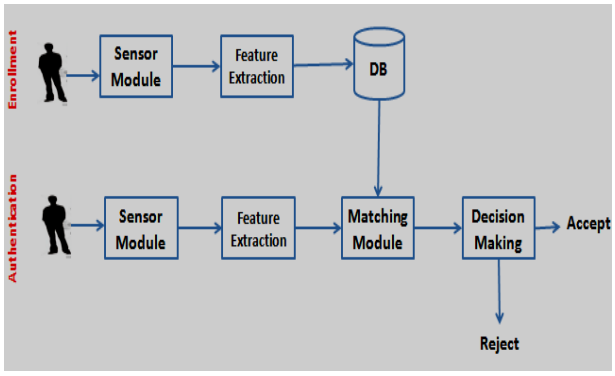

Figure 1. Authentication Biometric System Modules

A group of users was selected to participate in our experiments. The only requirement was that the participant is actually a user of a smart phone with a touch screen. Each user were asked to enter whatever message he wants (e.g. a description message about himself) in a single session.

In collecting data, we used two different techniques:

1. Few-training dataset.
2. More-training dataset.

In the first technique, 17 participants were asked to record 5 sessions. For each participant, three sessions would be used in the training phase. The remaining two sessions would be as testing sessions.

In the second technique, a group of 18 users participated in the experiments. A twenty sessions will be recorded for three volunteers of them. Here, 15 sessions will be used for training and the rest 5 sessions for testing. The remaining participants will be asked to record only 5 sessions for testing purposes.

Actually, dealing with a touchable keyboard is different from dealing with the hardware one. We found that a single touch gesture (tap, scroll, zoom ….etc) on the touch screen contains many of touch events. When the user touches the touch screen, the gesture starts and a stream of events take place. The gesture ends when the user's finger leaves the screen. Throughout this interaction, the android system tracks the position and other information of user's finger until the gesture ends. A sample of raw data collected from the touch screen and recorded by the phone is shown in Table I. The table shows the information about three taps on the touch screen (tap touch gesture). A single tap starts with action down and ends with action up. During this interaction, all finger's touch information is detected under the action move mode. That information are [13, 14]:

- The Down Time which is the time when the gesture was started in milliseconds (ms).
- The Event Time which is the time of the current touch event in milliseconds (ms).
- The size of the finger touch contacts the touch screen. The actual value in pixels is normalized and scaled to a value between 0 and 1.
- The finger pressure applied to the touch screen; or in other words, the strength of the finger touch contacts the touch screen. The pressure ranges from 0 (no pressure) to 1 (normal pressure).

C. *Feature Extraction*

We extracted the same features as in [5, 6]. Those features are:

- Hold time is the time between key-down and key-up of a single character.
- Up-Down time is the time between key-up of a character and key-down of the next character.
- Down-Down time is the time between key-down of a character and the key-down of the next character.

TABLE I. SAMPLE OF TOUCH RAW DATA COLLECTED USING OUR ANDROID APPLICATION

	Action Type	Down Time	Event Time	Size	Pressure
First Tap	DOWN	96274361	96274361	0.24705884	0.266666
	MOVE	96274361	96274424	0.27450982	0.3
	MOVE	96274361	96274458	0.19862746	0.333333
	UP	96274361	96274464	0.19862746	0.333333
Second Tap	DOWN	96276665	96276665	0.27450982	0.3
	MOVE	96276665	96276777	0.27450982	0.333333
	UP	96276665	96276786	0.21960786	0.333333
Third Tap	DOWN	96277247	96277247	0.19862746	0.266666
	UP	96277247	96277341	0.19862746	0.266666

The CMU dataset [7] contains 31 features extracted from 10-character password ***.tie5Roanl*** plus the return key. We extracted features for the first 11 characters typed by the participant. No specific text should be written by the participant (dynamic text).

For 11 characters, there are 11 of holds, 10 Up-Downs, and 10 Down-Downs. Total of 31 features will be extracted. We will extract two extra features which are average size and average pressure. Those features will be used later to investigate if they really have an effect on the authentication accuracy. All features will be extracted from both datasets.

D. Median Vector Classifier

The user authentication method involves two phases [12]: Training and Testing phases. In the Training phase, a user profile is built to be used later. In the testing phase, the user's typing behavior is compared to the stored user's profile to make a decision whether accept or reject that user.

We used the legitimate-attacker procedure as in [6]. The dataset will be composed into:

- Legitimate user training data.
- Legitimate user testing data.
- Attacker testing data.

In the first dataset (few-training data), through all 17 participants, one is selected as a legitimate user while the remaining 16 participants as attackers. All participants will do 5 sessions. For a selected legitimate user, 3 sessions are used in training phase to train the classifier and build a legitimate user profile (template) which will be used later in decision making in the testing phase. The remaining 2 sessions of the selected user are used as a legitimate user testing data. For the other 16 attackers, their 5 sessions are treated as attackers data (Fig. 2 illustrates this process).

In the training phase, the medians vector (μ) for the training set will be calculated (5 medians). In addition, the standard deviations vector (σ) for the same training set is calculated [6].

Later, the legitimate user and attackers testing data are tested depending on these two vectors (median and standard deviation).

In the testing phase, the standard deviation (σ) of the feature element has been selected to be the proximity distance threshold from the median (μ). For every feature element *i* from the 31 features; if the feature is within the proximity distance from the feature's median, mark the Feature Score (FS) as 1 otherwise mark it as 0 [6] as in (1).

$$FS_i = \begin{cases} 1, & f_i \in [\mu_i \pm \sigma_i] \\ 0, & f_i \notin [\mu_i \pm \sigma_i] \end{cases}, i = 1, 2, 3, \ldots F \quad (1)$$

Then, calculate the Test-Score (TS) as the sum of Feature-Scores (FS) as in (2).

$$TS = \sum_{i=1}^{F} FS_i \quad , i = 1, 2, 3 \ldots . F \quad (2)$$

After that, by determining the minimum percentage of accepted features which is the Pass-Threshold (PT), we will find the Pass-Score (PS) which is the minimum number of accepted features of a user as in (3).

$$PS = PT \times F \quad (3)$$

Finally, if the Test-Score is greater than Pass-Score (TS ≥ PS) then mark the input test as a legitimate (accepted input) and as an attacker (rejected input) otherwise.

The same steps will be done for the second type of dataset (more-training data). So among 18 users, three volunteers were selected to record 20 sessions. Here for a selected legitimate volunteer, 15 sessions will be used for training and the rest 5 sessions for testing. The remaining participants will be asked to record only 5 sessions to be used in the testing phase (Fig. 3 illustrate this process).

IV. System Evaluation

A. Evaluation Techniques

The performance of any biometric system is described by False Acceptance Rate (FAR) and False Rejection Rate (FRR) [12]. FAR is the probability of accepting an attacker while the FRR is the probability of rejecting a legitimate user. The point where FAR equals FRR is an Equal Error Rate (EER).

Systems that have lower FAR and FRR are more secure and more accurate in distinguishing users.

FAR and FRR are calculated using the following formulas (4,5):

$$FAR = \frac{Number\ of\ accepted\ AA}{Total\ number\ of\ AA} \quad (4)$$

$$FRR = \frac{Number\ of\ rejected\ LA}{Total\ number\ of\ LA} \quad (5)$$

were AA is an Attacker Attempts and LA is a Legitimate user Attempts.

In our proposed work, we test with different Pass-Thresholds (the minimum percentage of accepted features) ranging between 40% to 80% to find a suitable Pass-Threshold where that FAR is equal to FRR (EER). The value of PT can be tuned to reduce or increase the FAR and FRR depending on the access control priority. Reducing the FAR causes an increasing in rejecting a legitimate user (increase security and decrease usability). In the other hand, reducing the FRR would decrease the probability of rejecting a legitimate user and the probability of accepting an attacker is increased (decrease security and increase usability).

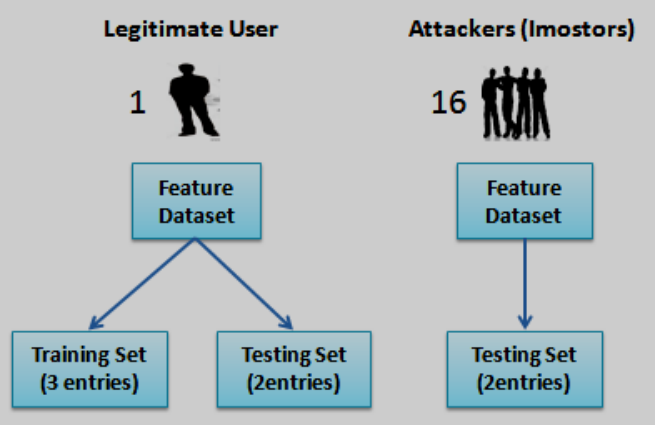

Figure 2. Dividing First Dataset (Few-Training Dataset) into Training and Testing Sets

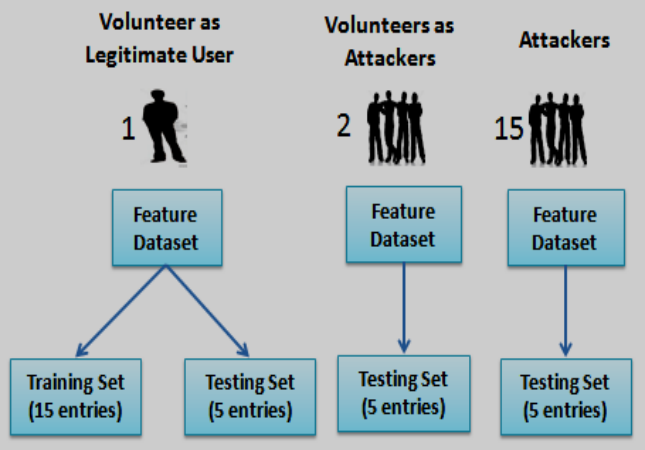

Figure 3. Dividing Second Dataset (More-Training Dataset) into Training and Testing Sets

B. System Results

By implementing the Median Vector Proximity algorithm [6] on the 31 extracted features from both datasets (few-training and more-training dataset), the FAR and FRR for different Pass-Thresholds are shown in Table II. The Pass-Threshold that results on FAR=FRR is about 45.9% for the few-training dataset and about 50.19% for the more-training dataset.

Also, the FAR and FRR for different Pass-Thresholds are calculated for the 33 features (adding the average of size and pressure features) as in Table III. We found that the EER (FAR and FRR are equal) when the Pass-Threshold = 46% for the few-training dataset and the Pass-Threshold = 51.71% for the more-training dataset.

The FAR and FRR with different Pass-Thresholds for both 31 and 33 features based on the Median Vector Proximity are shown in Fig. 5 and 6 for the few-training dataset and in Fig. 7 and 8 for the more-training dataset. We can see that when reducing Pass-Threshold, the FAR will be increased (less secure system). In the other hand, when increasing the Pass-Threshold, FRR will be increased (more secure system). As we can see, the FAR and FRR for all systems are semi-equal when the PT=50%.

In the few-training dataset, the EER average and EER standard deviation is calculated for all 17 typists when applied on 31 and 33 features. In the more-training dataset, the average and standard deviation is calculated for the three volunteers (each time one of them will be selected as a legitimate user). All EER average and standard deviation results are shown in Table IV.

As we can see from Table IV, when using 33 features the avgEER is decreased by 0.7% in the few-training dataset and by 0.37% in the more-training dataset; which brings better results. This shows that the accuracy of the system will be increased by the increasing in the features set. We can figure that the average EER in the more-training dataset is decreased by 12%. This shows that this classifier (Medians Vector Proximity) becomes more accurate when increasing the training data. Also, by increasing the pass threshold over 50%, the system will become more secure but less usable. So depending on the user's needs, we can tune the pass threshold to achieve a desirable result.

TABLE II. FAR AND FRR FOR DIFFERENT PASS-THRESHOLDS USING MEDIAN VECTOR PROXIMITY (31 FEATURES)

		Pass Threshold				
		40%	50%	60%	70%	80%
Few-training dataset	FAR	0.5404	0.2206	0.0784	0.0123	0.0037
	FRR	0.1176	0.2353	0.4118	0.6667	0.8431
More-training dataset	FAR	0.1255	0.0078	0.0039	0.0039	0
	FRR	0	0	0.1333	0.4667	0.6667

TABLE III. FAR AND FRR FOR DIFFERENT PASS-THRESHOLDS USING MEDIAN VECTOR PROXIMITY (33 FEATURES)

		Pass Threshold				
		40%	50%	60%	70%	80%
Few-training dataset	FAR	0.5025	0.1985	0.0735	0.0147	0.00037
	FRR	0.0784	0.2353	0.4314	0.6667	0.8235
More-training dataset	FAR	0.0902	0.0078	0.0039	0.0039	0
	FRR	0	0	0	0.4000	0.6000

TABLE IV. THE AVERAGE EER AND STANDARD DEVIATION EER FOR OUR PROPOSED WORK

# Features	Few-Training Dataset		More-Training Dataset	
	EER Avg.	EER Std.	EER Avg.	EER Std.
31 Touch Keystroke Features + 2 Touch Features	0.1219	0.1337	0.0039	0.0068
31 Touch Keystroke Features	0.1293	0.1361	0.0076	0.0132

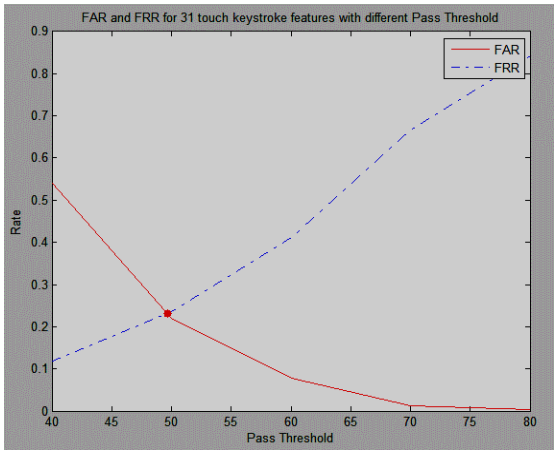

Figure 5. FAR and FRR for 31 touch keystroke features with different Pass Threshold (few-training dataset)

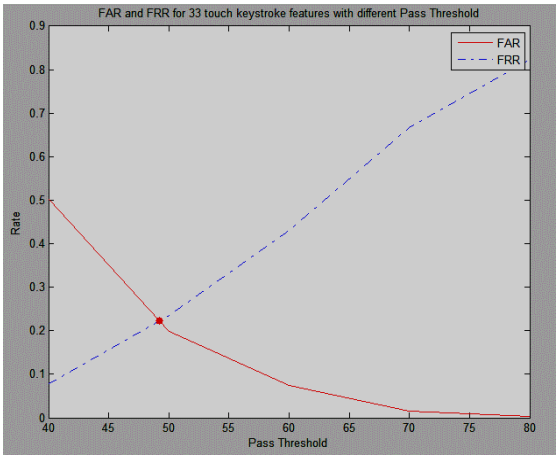

Figure 6. FAR and FRR for 33 touch keystroke features with different Pass Threshold (few-training dataset)

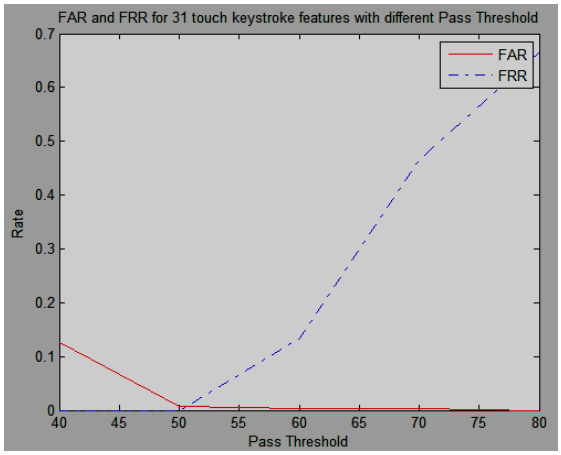

Figure 7. FAR and FRR for 31 touch keystroke features with different Pass Threshold (more-training dataset)

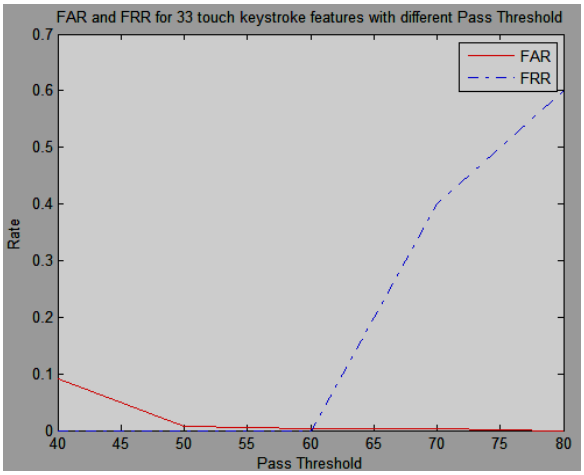

Figure 8. FAR and FRR for 33 touch keystroke features with different Pass Threshold (more-training dataset)

From Table V, the author in [6] applied his experiments on personal computers and had got EER=8%. The remaining literature used features that are based on the touch screen. In [3, 9] authors used long text which brings more features so they had the least EER (4.5 and 7.5%). Authors in [11] used the same password in [6] but they extract touch keystroke features besides some extra touch features. They had got an EER = 15.3% and EER=12.9% for 41 and 71 features respectively. Finally, our proposed work which did not depend on a specific text (using few-training dataset); had got EER=12.9% for purely 31 touch features and EER=12.2 for the same features set with two extra touch features (size and pressure). Using more-training dataset brings amazing results with EER=0.76% for 31 features and EER=0.39% for 33 features. Table V summarize this comparison between our proposed work and the literature.

V. CONCLUSION AND FUTURE WORK

There is an increasing demand for more secure access control in many of today's mobile applications. The touch screen is the primary input medium on the latest mobile phones [5]. The keystroke dynamics provides a natural choice for secure mobile access. Also, it is cost-effective since no extra hardware is needed; only suitable software is needed to collect keystroke timing information.

In this paper, we proposed to apply the Median Vector Proximity [6] classifier on the touch keystroke data derived from mobile's touch screen (touchable keyboard). We evaluated the system on no specific text (dynamic text) and extracted 31 and 33 touch features. When using few-training dataset, the average EER were 12.9% and 12.2% respectively. While using more-training dataset brings amazing results with EER=0.76% for 31 features and EER=0.39% for 33 features. This shows that this classifier (Medians Vector Proximity) becomes more accurate when increasing the training data. Also, we found that by increasing the number of features, the average EER was

reduced by about 0.7% in the few-training dataset and by 0.37% in the more-training dataset. Therefore, the more features we used results in more accurate systems. As a future work, we will apply different classifiers on those features and find out which classifier would give better results.

TABLE V. COMPARISON BETWEEN OUR PROPOSED WORK AND THE LITERATURE

Literature *(Algorithm)*	**Text**[a]	**Applied on**	**EER Avg.**	
Al-Jarrah [6] *(Median Vector Proximity)*	".tie5Roanl"	31 Hardware Keystroke Features	0.08	
Nan Zheng et al [4] *(Nearest Neighbor Algorithm)*	Some specified 4-digits and 8-digits PINs	Touch Keystroke + other Touch Features	0.0455	
Xuan Huang et al [10] *(Statistical Approach)*	Username="abertaytest" password="abertay2011"	42 Touch Keystroke Features	0.075	
Margit Antal et al [11] *(Manhattan)*	".tie5Roanl"	41 (Only Touch Keystroke Features)	0.1530	
		71 (Touch Keystroke Features + other Touch Features)	0.1290	
Our Proposed Work *(Median Vector Proximity)*	Not specified (any text could be used)	31 Touch Keystroke Features + 2 Touch Features	0.1219 (few-training dataset)	0.0039 (more-training dataset)
		31 Touch Keystroke Features	0.1293 (few-training dataset)	0.0076 (more-training dataset)

a. The TEXT that is used in the researcher's experiments

ACKNOWLEDGMENT

This research has been supported by King Abdulaziz City for Science and Technology (KACST), under grant number (PGP – 35 - 444). The authors, therefore, acknowledge with thanks, KACST technical and financial support.

REFERENCES

[1] Alghamdi, S.J. and L.A. Elrefaei, *Dynamic User Verification Using Touch Keystroke Based on Medians Vector Proximity*, in *7th International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN)*. 2015, IEEE. p. 121-126.

[2] Femila, M.D. and A.A. Irudhayaraj. *Biometric system*. in *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*. 2011. IEEE.

[3] Shanmugapriya, D. and G. Padmavathi, *A survey of biometric keystroke dynamics: Approaches, security and challenges.* arXiv preprint arXiv:0910.0817, 2009.

[4] Zheng, N., et al. *You are how you touch: User verification on smartphones via tapping behaviors*. in *Network Protocols (ICNP), 2014 IEEE 22nd International Conference on*. 2014. IEEE.

[5] Feng, T., et al. *Continuous mobile authentication using touchscreen gestures*. in *Homeland Security (HST), 2012 IEEE Conference on Technologies for*. 2012. IEEE.

[6] Mudhafar, M., *Al-Jarrah, An Anomaly Detector for Keystroke Dynamics Based on Medians Vector Proximity.* Journal Of Emerging Trends In Computing And Information Sciences VOL3, 2012.

[7] Killourhy, K.S. and R.A. Maxion. *Comparing anomaly-detection algorithms for keystroke dynamics*. in *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*. 2009. IEEE.

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

[9] Frank, M., et al., *Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication.* Information Forensics and Security, IEEE Transactions on, 2013. **8**(1): p. 136-148.

[10] Huang, X., G. Lund, and A. Sapeluk. *Development of a typing behaviour recognition mechanism on android*. in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. 2012. IEEE.

[11] Antal, M., L.Z. Szabó, and I. László, *Keystroke Dynamics on Android Platform.* 2014.

[12] Jain, A.K., A. Ross, and S. Prabhakar, *An introduction to biometric recognition.* Circuits and Systems for Video Technology, IEEE Transactions on, 2004. **14**(1): p. 4-20.

[13] *Android Developers - Touch Devices*. [cited 2015 3/2/2015]; Available from: https://source.android.com/devices/input/touch-devices.html.

[14] *Android Developers - Motion Event*. [cited 2015 3/2/2015]; Available from: http://developer.android.com/reference/android/view/MotionEvent.html.