2016 IEEE Security and Privacy Workshops

Improving Performance and Usability in Mobile Keystroke Dynamic Biometric Authentication

Faisal Alshanketi
ECE Department
University of Victoria
P.O. Box 1700 STN CSC
Victoria, B.C. V8W 2Y2 Canada
Email: faisal_ fadel2@hotmail.com

Issa Traore
ECE Department
University of Victoria
P.O. Box 1700 STN CSC
Victoria, B.C. V8W 2Y2 Canada
Email: itraore@ece.uvic.ca

Ahmed Awad E. A
School of Eng. & Comp. Sciences
New York Institute of Technology
701 West Georgia Street
Vancouver, B.C. V7Y 1K8 Canada
Email: Ahmed.Awad@nyit.edu

***Abstract*—In the last few years, the number of mobile devices such as smartphones and tablets, in circulation, has increased dramatically. The primary and often only protection mechanism in these devices is authentication using a password or a Personal Identification Number (PIN). Passwords are notoriously known to be a weak authentication mechanism, no matter how complex the underlying format is. A more secure alternative option which has gained interest recently is extracting keystroke dynamic biometrics from supplied passwords for mobile authentication. In this paper, we show that using random forests classifier, improved accuracy performance can be achieved for mobile keystroke dynamic biometric authentication. We also propose a new algorithm for handling typos, which is an essential step in improving usability. We study both timing features and pressure-based features. Experimental evaluation is based on two public datasets and a third dataset collected in our lab. The best performance, obtained by combining timing and pressure features, is an Equal Error Rate (EER) of 2.3% for a population of 42 users.**

***Keywords*—Keystroke Dynamics, Mobile Security, Mobile Authentication, Biometric Authentication, Typo Handling.**

I. Introduction

The last decade has seen a dramatic increase in the number and sophistication of mobile devices. Mobile devices are increasingly used to perform complex and sensitive computations and store private data and resources. However, the protection available for these devices remains still at the bare minimum level.

The main protection for most devices consists of a simple password or a Personal Identification Number (PIN). However, passwords are known to be a weak form of authentication, no matter how complex the password string is. Passwords can be hacked, stolen, forgotten, or shared.

Recently, we have seen a growing interest in using biometric technologies as alternatives or reinforcement for passwords. Among the technologies being used or under consideration is keystroke dynamic biometrics.

The roots of keystroke biometrics go back to the World War II, where military intelligence used a methodology called ”The Fist of the Sender” to distinguish between a Morse Code message sent by ally or enemy operators [1]. Today keystroke dynamics is a well known behavioural biometric technology that has several benefits over other biometric technologies. For instance, keystroke dynamics biometric can be used in both static and continuous authentication, it is a resettable biometric technology, it operates in indoor and outdoor environments, it does not require special hardware sensor, and it can be used unobtrusively.

Over the last two decades much work has been done on using keystroke dynamics data collected from conventional computer keyboard for user authentication.

Recently, several proposals have been published on applying keystroke dynamic biometric for mobile authentication. Most of the proposals use the standard keystroke features, namely, dwell time (the time between pressing and releasing a single key) and the flight time (the time between two consecutive key presses). Some of the challenges encountered in this process are related to the differences in environments and platforms. For instance, some of the keystroke features that are readily available on conventional keyboards, cannot always be captured in current mobile platforms.

As a consequence, the accuracies achieved with conventional keystroke authentication have not always been translated in the mobile keystroke authentication proposals published so far. The error rates obtained for mobile devices remain on average relatively high.

Recently, it has been shown that better accuracy can be achieved using new features extracted from the finger pressure. Despite the encouraging results achieved with these new features, there is still room for significant improvement in accuracy.

Another key limitation of the previous work on using keystroke dynamics for mobile authentication is related to typo handling or the lack thereof. For example, if the user by mistake typed a wrong character the entire trail or login attempt would be rejected biometric authentication. While this might be acceptable in implementing keystroke in desktop environment, it creates a major usability issue in mobile platform, where users can easily hit the wrong key on a regular basis. Not giving users the ability to correct typos and asking them to retype their credentials (e.g. user name and password) discourage from using strong password formats (e.g. mix of characters, digits, upper-cases, lower-cases, and special characters).

DOI 10.1109/SPW.2016.12

We present in this paper a new approach for mobile keystroke dynamic biometric authentication, that tackles the aforementioned challenges. Specifically we show that improved performance can be achieved by tuning adequately a random forests classifier. We consider the different group of features mentioned above (i.e. standard and pressure-based), and evaluate the proposed model using 3 different datasets involving cumulatively 103 users. The three datasets consist of 2 public datasets of 51 and 42 users, respectively, and a third dataset of 10 users collected in our lab. We also introduce a new algorithm for typo handling, and evaluate the effectiveness of the algorithm using the dataset collected in our lab.

The rest of the paper is structured as follows. In section II, we summarize and discuss related work on keystroke dynamics for mobile authentication. In Section III, we describe our feature space and classification model for mobile authentication. In section IV, we describe the experimental evaluation of the proposed approach. In Section V, we present our approach for handling typing errors on keystroke dynamics for mobile authentication. In Section VI, we make concluding remarks and summarize future work.

II. Related Work

Several papers have been published in the literature on keystroke dynamic biometric authentication for conventional computing devices. However, only a relatively smaller amount of works have focused so far on keystroke biometrics for soft keyboards on mobile devices.

In 2006, Clarke and Furnell conducted the first experiment on keystroke dynamic analysis for mobile devices [2]. In the experiment, 30 participants were asked to type for 30 times two different password strings, of size 4 and 11 digits, respectively. The authors extracted as features the key dwell time and digraph flight time which were processed using neural network classifiers. On average, an equal error rate (EER) of 12.8% was obtained.

Karnan and Krishnaraj conducted a comparison between different biometrics techniques [3]. In the study a dataset consisting of 200 samples for keystroke, 100 for finger print and 100 for palm print, were used. They used Particle Swam Optimization (PSO), Ant Colony Optimization (ACO) and Bacteria Foraging Algorithm (BFOA) with Support Vector Machine (SVM) classifier to improve the performance. For the keystroke, they extracted as features the dwell time and flight time for digraphs, and computed the mean and standard deviation to create biometric templates. They selected finger print and palm print as the input to the BFOA algorithm to discover the most important subset of features. By applying SVM classification with BFOA to the individual biometric factors separately, error rates[1] of 0.069%, 0.076% and 0.083% were obtained for Keystroke, Fingerprint and Palmprint, respectively. The best result was obtained for the combination of the three modalities (Keystroke, Fingerprint, Palmprint) where they achieved 92.8% accuracy in detecting imposters with 0.063% being the error rate obtained for the BFOA algorithm. However, they did not mention how the data was collected and how many subjects were involved in the study.

Maiorana and Campisi [4] proposed an approach for user authentication using keystroke dynamic biometric for mobile devices.

An experiment was conducted to collect keystroke dynamic samples, where 40 participants were asked to type 6 different passwords 20 times each, giving a dataset of 4800 samples. Two features were extracted from the raw data consisting of the dwell time and flight time. The extracted features were processed by varying the number of keys and using different classifiers, specifically, fuzzy C-means, Bayes Classifier, Support Vector Machine (SVM), as well as Principal Component Analysis (PCA). The best performance achieved was an EER of 13.59% with 10 keystrokes long passwords obtained using fuzzy C-means. Among the limitations of the proposed approach are the relatively high EER especially considering the length of the password string.

Karnan and Krishnaraj [5] applied Ant Colony Propagation(ACP) with Back Propagation Neural Network (BPNN) for keystroke biometric authentication in mobile devices. They collected a dataset with 25 participants, each providing 50 samples. Two types of features were extracted, namely, dwell time and flight time. The comparison between ACP and Genetic algorithm produced classification error rates of 1.07% and 0.20% where the accuracy was 88.9% and 88.6% of combined features. On the other hand, Particle Swam Optimization (PSO) produced a classification error rate of 0.006% and accuracy of 94.8%.

Bours and Masoudian [6] claimed their work to be the first to investigate keystroke dynamics on randomly generated one-time PIN (OTPs). Data was collected from 30 subjects where each subject typed 6 digits OTPs 150 times correctly (without typo). An EER of 26% was obtained, which is relatively high.

Saevanee and Bhattarakosol [7] presented an approach for mobile authentication using keystroke dynamic where in addition to the standard features (i.e. dwell and flight times), the impact of the finger pressure was investigated. They conducted an experiment by collecting data for 10 users where each user typed 30 times their cell phone numbers consisting of 10 digits. Three features were extracted consisting of the dwell time, the flight time, and the finger pressure. By applying neural network classification, to the individual features separately, EER of 35%, 40% and 1% were obtained for the dwell time, flight time, and finger pressure, respectively. The combination of the dwell time, flight time, and finger pressure produced an EER of 9% while using just the dwell time and flight time produced an EER of 29%. They established that using the finger pressure is better than using the combination of dwell time and flight time to identify the genuine user. An accuracy rate of 99% was obtained for the combination of the dwell time and the finger pressure; interestingly the same accuracy was obtained for the finger pressure alone. The main issue with the proposed work is the relatively small size of

[1] Note that the error rate is a vague metric in the context of biometric, and should not be equated with the EER.

the validation population and also the fact that participants provided samples by typing different set of characters (i.e. individual phone numbers), which creates some confounds.

Another study using finger pressure with dwell time and flight time as features for keystroke biometric authentication on mobile devices was conducted by Sen and Muralidharan [8]. Data was collected from 10 subjects where each participant typed a 4-digits password 100 times in 5 sessions with 20 samples in each session. They applied four different classifiers, a decision tree classifier - J48, a Naive Bayes classifier, the K* classifier and a multilayer perceptron classifier. The best performance was an EER of 15.2% obtained using the multilayer perceptron classifier. The main limitation with this study is similar to Saevanee's.

In addition to the standard features (i.e. dwell and flight times), Trojahn [9] investigated finger pressure and size of the finger for mobile authentication using keystroke dynamic. The author conducted an experiment where 152 participants were asked to type a predefined 17-digits password 10 times in a single session. By applying K-means classification, to the individual features separately, FAR of 8.03%, 12.66% and 12.63% and FRR of 12.3%, 11.64% and 33.33% were obtained for the dwell time, flight time, and trigraph (three different keys), respectively. The combination of the dwell time, flight time, finger pressure and the size of the finger produced a FAR of 4.19% and a FRR of 4.59% which are very encouraging results. While using just the dwell time and flight time, a FAR of 9.28% and a FRR of 6.72% were obtained. While the authors avoided some of the experimental mistakes achieved in the previous works using finger pressure, the main limitation is the fact that the data was all collected in a single session.

Draffin, Zhu, and Zhang [10] applied Back Propagation Neural Network (BPNN) for keystroke biometric authentication in mobile devices. Data was collected during a period of 3 weeks from 13 subjects. Beside dwell time and flight time features, finger pressure, finger area, drift, and device orientation were extracted. A FAR of 14.0% and a FRR of 2.2% were obtained as performance. They did not explain the mechanism of collecting their data, which raises more unanswered questions.

A recent study using finger pressure and finger area as features for keystroke biometric authentication in mobile devices was presented by Antal and Szabo [11]. They asked 42 participants to type the same password (**.tie5Roanl**) 30 times in 2 sessions. Four features were extracted consisting of the dwell time, the flight time, the finger pressure and the finger area. They applied the same script R developed by Killourhy and Maxion [12]. The script R implements three anomaly detection methods based on Euclidean, Manhattan, and Mahalanobis distances. The best performance achieved was an EER of 12.9% with pressure and finger area features obtained using Manhattan distance while an EER of 15.3% was achieved using the standard features (i.e. dwell and flight times).

III. Features Space and Classification Model

In this study we will consider both the standard features (dwell and flight times) and the pressure features (finger pressure and finger size). For the flight time, we consider the following variations: release-to-press (**RP**) (the duration of the time interval between a key released and a key pressed), press-to-press (**PP**) (the duration of the time interval between two keys pressed), release-to-release (**RR**) (the duration of the time interval between two keys released).

We use Random Forest to classify individual users and discriminate genuine users from impostors.

Random forest is a machine learning technique also called bagging because it combines multiple learning models to increase the classification accuracy. It is a classification and regression method that generates random subsets of data and variables, from which multiple decisions trees are created. It is an ensemble technique which takes a decision based on majority vote of classification prediction.

IV. Experiments

A. Datasets

We used in our experimental evaluation three different datasets. Two of the datasets are public datasets, while the third dataset was collected in our lab.

1) Dataset 1: The first dataset used in our work (Dataset 1) provides only keystroke timing data, i.e., only standard keystroke features (dwell and flight times) can be extracted from the data. The dataset was collected by El-Abed, Dafer and El Khayat from Rafik Hariri University, and made available online at http://www.coolestech.com/RHU-Keystroke/. The data was collected from 51 subjects who typed the same password string "rhu.uvinversity" between 15-20 times, in 3 sessions spread over 3-30 days [13]. The dataset contains 985 samples where the minimum number of samples per user is 15 samples and the maximum number of samples per user is 21 samples. The average number of samples per user is 17 samples.

2) Dataset 2: The second dataset (Dataset 2) provides both timing and pressure data from keystrokes. The dataset was collected by Antal and Szabo from Sapientia University, and made available online at http://www.ms.sapientia.ro/∼manyi/keystroke.html. The data was collected from 42 subjects who typed the same password string ".tie5Roanl" 51 times, in at least 2 sessions spread over 2 weeks [11].

3) Dataset 3: The third dataset (Dataset 3) was collected in our lab with the intent of studying the impact of typo handling on performance. We developed a client/server application to collect data from participants. The mobile client application used for data collection is a hybrid application written in Javascript and HTML 5, while the server is a lightweight server written in python and is based on the Tornado web framework. TinyDB NoSql database was used smartphone in the back-end. The data was collected using Galaxy SIII from 10 participants, who typed 30 times (spread over 2 sessions) the password string [Mohammed-63]. We allowed typo, for example, if the user by mistake typed a wrong character the entered trail or login attempt is accepted.

B. Data Analysis Techniques

We applied Random Forest algorithm [14] in our experiment to classify the genuine users and identify imposters. A profile is built for each user using a training set consisting of positive or genuine samples from the user, and negative or imposter samples from other users.

We used 10-fold cross validation, where 90% of the dataset is used for training and 10% for testing. In each test round, we calculate the FRR individually for each user, by comparing the (10%) genuine test samples (i.e. from the user) against the user's profile. Similarly, the FAR is calculated for each user by comparing the (10%) test samples from other users against the user's profile. The FAR/FRR for the test round is obtained by averaging the individual FAR/FRR. Furthermore the overall FAR/FRR are calculated by averaging the values obtained over the 10 rounds of cross validation.

In the above evaluation approach, the imposter class has more samples than the genuine class, which will generate imbalance class distribution. The potential outcome of imbalance class distribution is what is known as a majority classifier, where there is a tendency of classifying all samples as belonging to the majority class (i.e. impostor samples). The approaches commonly used in the literature to handle imbalance class distribution include cost sensitive learning and sampling.

Cost sensitive learning assigns weights to the training samples expressing different misclassification costs. Specifically, positive (or genuine) samples are assigned greater weight compared to the negative (impostor) ones.

Sampling involves under-sampling the majority class (i.e. impostor) or over-sampling the minority class (genuine). Under-sampling consists of selecting for training a subset of the majority class, while over-sampling consists of using for the same purpose a superset of the minority class. The superset can be generated by reusing multiple times the same samples or by generating synthetic samples.

In this work, we used in our evaluation experiments both cost sensitive learning and under-sampling. We perform cost sensitive training by assigning a weight P (denoted weight(P)) to the imposter class corresponding to the ratio between the total number of genuine samples and the total number of imposter samples.

C. Data Analysis and Results

We started the evaluation with Dataset 1 using cost sensitive learning. Fig. 1 and Table I depict the ROC curve and sample performance results, respectively, obtained by varying weight(P). The best result was obtained when setting the weight (P) to 0.000515, corresponding to EER = 5.8 %.

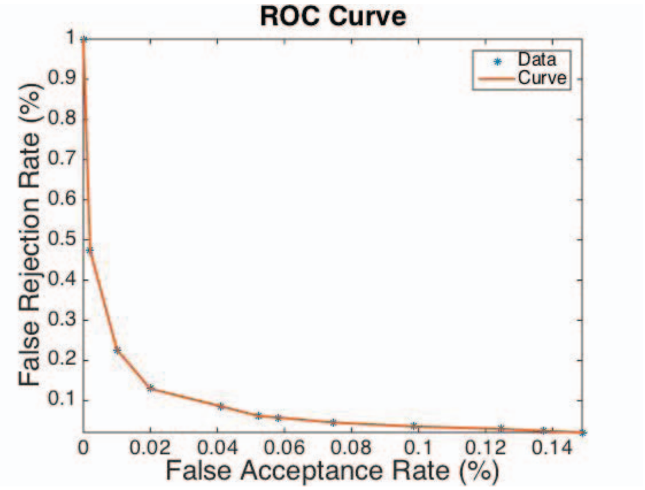


Fig. 1. ROC curve without under sampling the negative class and by varying the weight (P) for dataset 1.

TABLE I
PERFORMANCE OBTAINED FOR DATASET 1 WITHOUT UNDER SAMPLING THE NEGATIVE CLASS (ALL SAMPLES = 985) AND VARYING THE WEIGHT (P).

Weight(P)	FRR %	FAR %
0	0	1
0.0001	0.2	47.4
0.0002	1	22.7
0.0003	1.99	12.97
0.0004	4.08	8.64
0.0005	5.24	6.14
0.000515	**5.8**	**5.8**
0.0006	7.44	4.55
0.0007	9.85	3.57
0.0008	12.47	2.95
0.0009	13.73	2.45

In addition to cost sensitive learning, we applied under-sampling to Dataset 1. By varying weight(P) and using under-sampling, the best performance was obtained when setting weight(P)=0.0024. Let u denote our under-sampling parameter, which in other words represents a percentage of the negative samples available for training.

Fig. 2 and Table II show the ROC curve and sample results, respectively, when setting the weight(P) = 0.0024 and varying u from 18 samples to the maximum count of the negative samples. The best result is obtained for u = 380 as EER = 6.6 %.

This indicates that with under-sampling, there is a slight decrease in accuracy. But at the same time the lower amount of training data involved means lower computation cost.

Next, we applied our algorithm to Dataset 2 by conducting three different experiments and varying weight(P). In the first experiment, we considered only the pressure features (finger pressure and finger area). The second experiment used only the timing features (dwell and flight times), while the third experiment considered all the above features. Fig. 3-5 depict the ROC in these experiments; table III lists the obtained error rates. The curves show the relation between the FAR and FRR when varying weight(P) from 0.0001 to 0.0009 and using the maximum number of negative samples (i.e. without under-

TABLE II
PERFORMANCE OBTAINED BY UNDER SAMPLING THE NEGATIVE CLASS AND VARYING THE MAXIMUM COUNT OF THE NEGATIVE SAMPLE FOR DATASET 1.

under sampling (# samples)	Weight(P)	FRR %	FAR %
18	0.0024	0	1
180	0.0024	0.6	32.72
280	0.0024	2.72	13.44
380	0.0024	**6.6**	**6.6**
480	0.0024	11.21	4.16
580	0.0024	15.72	2.81
680	0.0024	19.39	2.01
780	0.0024	23.48	1.48
880	0.0024	25.26	1.26
full dataset (985)	0.0024	30.6	0.49

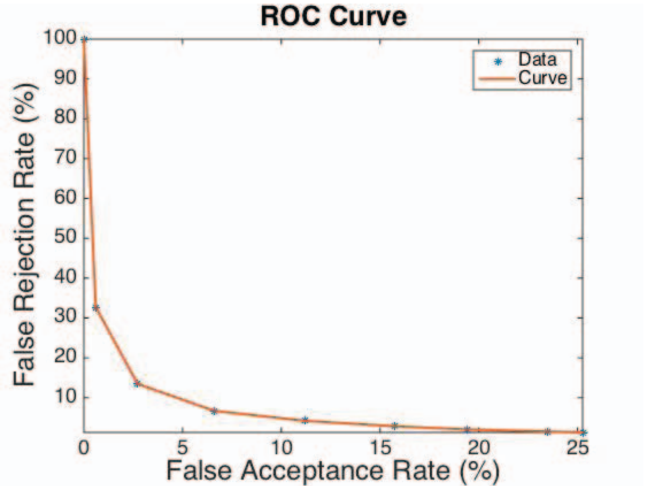


Fig. 2. ROC curve when under sampling the negative class and making the weight (P) constant for dataset 1.

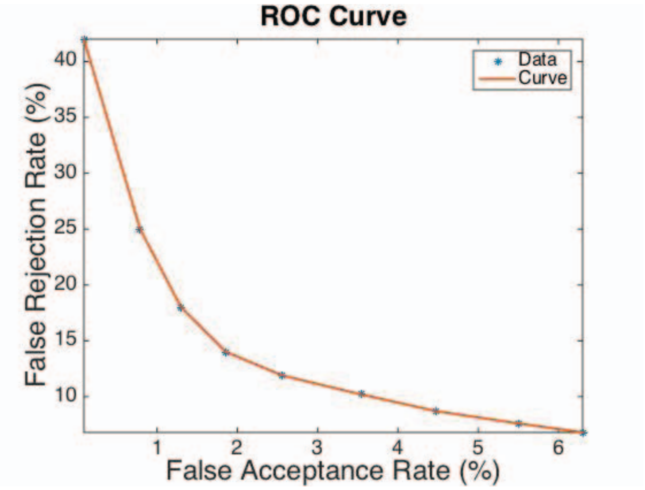


Fig. 3. ROC curve using just pressure and finger size and varying the weight (P) for dataset 2.

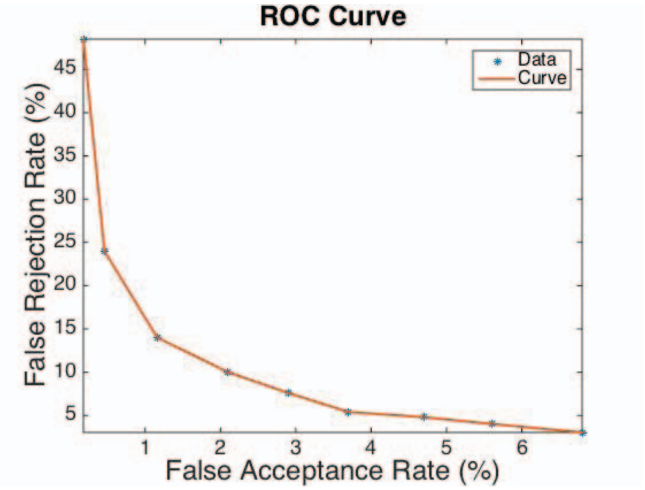


Fig. 4. ROC curve using just Dwell Time (DT) and Flight Time (FT) and varying the weight (P) for dataset 2.

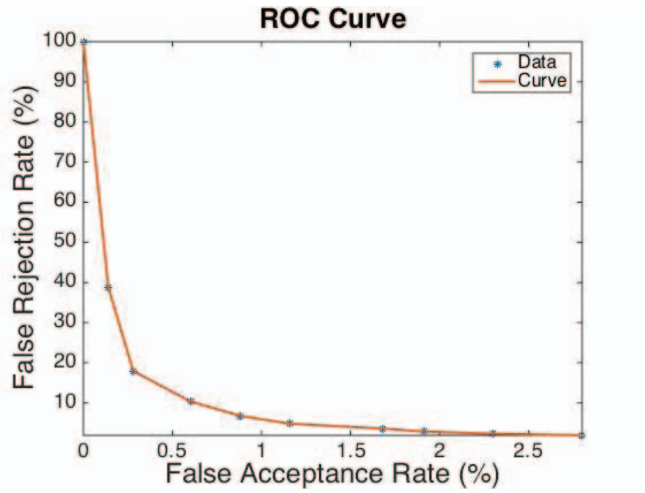


Fig. 5. ROC curve using FT, DT, pressure and finger size and varying the weight (P) for dataset 2.

sampling). The best result was obtained with the full features set (i.e. both timing and pressure) when setting the weight (P) to 0.0008 corresponding to EER = 2.3 %. On the other hand, using dwell time and flight time an EER of 6.5% was achieved, and while using just pressure and finger area an EER of 4.7% was achieved.

From these results, it is clear that pressure features perform better than timing features. Furthermore, as expected, the combination of all the above feature categories provide much better performance compared to using only a single category.

Table IV summarizes and compares with our work the performances obtained for existing work on keystroke dynamics biometric for mobile devices. Our proposed method performs well comparatively with the existing work. This is encouraging considering that a varied dataset was used.

TABLE III
THE RESULT OBTAINED BY APPLYING RANDOM FOREST FOR DATASET 2.

Features	Weight(P)	FRR %	FAR %
DT + FT	0.0007	4.7	4.8
Pressure + finger area	0.0009	6.3	6.8
DT + FT + Pressure + finger size	0.0008	2.3	2.3

TABLE IV
COMPARISON OF EXISTING RESEARCH WORKS WITH OUR WORK

References	Number of Subjects	Features	Algorithms	Result
[2]	30	dwell time and digraph flight time	Neural Network	EER=12.8%
[3]		dwell time and digraph flight time	PSO, ACO and BFOA with SVM	ER=.063
[4]	40	dwell time and digraph flight time	Bayes, SVM PCA	EER=13.59%
[5]	25	dwell time and digraph flight time	PSO and ACP with BPNN	ER=0.006
[6]	30	dwell time and digraph flight time	SMD, Adpted (SMD), SED, ASED	EER=26%
[7]	10	dwell time, digraph flight time and finger pressure	Probabilistic Neural Network	EER=9%
[8]	10	dwell time, digraph flight time and finger pressure	Multilayer Perception	EER=15.2%

[9]	152	dwell time, digraph flight time, finger pressure and size of the finger	K-means	FAR=4.19%
				FRR = 4.59%

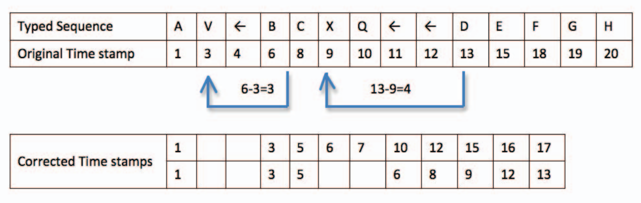
[10]	13	dwell time, digraph flight time, finger pressure, finger area, drift and device orientation	Back Propagation Neural Network	FAR=14.0%
				FRR = 2.2%

[11]	42	dwell time, digraph flight time, finger pressure and finger area	Manhattan Metrics	EER = 12.9%
This paper dataset 1	51	dwell time and digraph flight time	Random Forest	EER=5.8%
This paper dataset 2	42	dwell time, digraph flight time, finger pressure and finger area	Random Forest	EER=2.3%

V. HANDLING TYPING ERRORS

Typing errors occur whenever there is a misspelling of a word while typing passwords. In order to deal with such errors, we assume the following proposition.

Consider a case when a user mistyped his/her password and realizes that he/she mistyped it. To correct the password, the user deletes the wrong characters by pressing the backspace key and then completes the password afterwards. In doing so, the user types some extra characters which will also be logged in the time stamps registers. As a consequence, not only there will be some extra entries in the registers but the flight times between the correct digits would also differ. Fig. 6 shows an example of such a scenario.



Typed Sequence	A	V	←	B	C	X	Q	←	←	D	E	F	G	H
Original Time stamp	1	3	4	6	8	9	10	11	12	13	15	18	19	20

Corrected Time stamps	1			3	5	6	7	10	12	15	16	17
	1			3	5			6	8	9	12	13

Fig. 6. An illustration of a mistyped password.

Here the user typed three extra characters, V, X and Q, while typing the password ABCDEFGH. Also, after realizing the mistake, the user deletes these characters by pressing the backspace key which is shown as ← in the figure.

Our typo handling algorithm makes the assumption that the flight time between the incorrect letter and the preceding letter is approximately the same as the time between the correctly typed letters.

Based on this assumption, we compute the time stamps for the password with extra characters as follows. Firstly we find the difference between the time stamps of the incorrect letter and the corresponding correct letter (V and B in the Fig. 6), and then we subtract that difference from each of the timestamps of the correctly typed letter and the letters following it. For example, in the Fig. 6, the difference between the time stamps of letters V and B is 3 ms. Afterwards, the computed time difference is subtracted from the timestamps of the letters starting from the letter B. The first row of the 'Corrected Time Stamps' shows the updated time stamps. The same procedure can be extended to the typing errors caused by typing multiple extra characters as is the case with the letters (X and Q) in the Fig. 6. Here the correct time stamps are computed similarly by subtracting the time difference between the timestamps of the first incorrect letter and the corresponding correct letter. Notice that, by this procedure, we are able to compute the correct time stamps as given in the Table V.

We conducted a comparison between handling and not handling typing errors using Dataset 3 which was collected in our lab and includes typo data.We started by running our classification model on the dataset without removing the typos, and without applying our typo handling method. Next, we

TABLE V
AN ILLUSTRATION OF A CORRECT PASSWORD

Password	A	B	C	D	E	F	G	H
Time Stamp	1	3	5	6	8	11	12	13

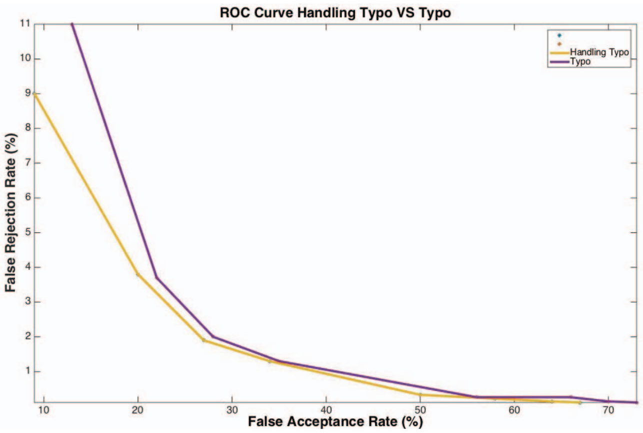


Fig. 7. ROC curves when handling typo vs. no typo handling and varying the weight (P) for dataset 3.

repeated the same process, by running on the same dataset, the classifier and the typo handling algorithm. Fig. 7 and Table VI show the obtained results. In the first case, EER=12% was obtained, while when applying our typo handling algorithm, an EER=9% is obtained. The improvement in performance is not surprising, as the typo handling is expected to correct the negative impact of typos.

On the other hand, typo handling can have adverse effect on accuracy. In order to assess the strength of our typo handling method, we ran an experiment on a clean version of dataset.In this dataset, we excluded the incorrect trails meaning that the typing mistakes were not allowed. For example, if a user typed a wrong character by mistake, the entire trail or login attempt would be rejected. Fig. 8 depicts the ROC curve obtained by varying weight (P). An EER of 9% is obtained which is almost the same result when we handled the typo mistakes. This implies that our proposed scheme for handling typing mistakes does not degrade significantly the accuracy. With this approach, we are able to improve usability (by allowing typos) while maintaining the same level of accuracy.

TABLE VI
PERFORMANCE OBTAINED BY HANDLING TYPO VS. NO TYPO REMOVAL, WHEN VARYING THE WEIGHT (P) FOR DATASET 3.

Weight(P)	FRR % No Typo	FAR % No Typo	FRR % (Typo)	FAR % (Typo)
0 .24	67	0.11	73	0.11
0.23	64	0.14	70	0.14
0.22	58	0.22	66	0.26
0.21	50	0.33	56	0.26
0.042	34	1.3	36	1.3
0.032	27	1.9	28	2
0.022	20	3.8	22	3.7
0.0021	**9**	**9**	**13**	**11**

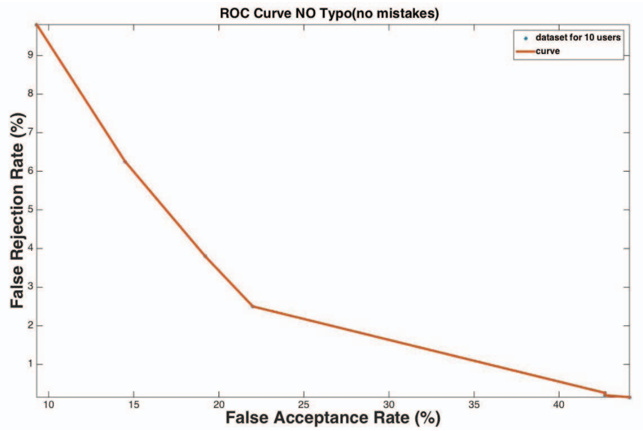


Fig. 8. ROC curves when no typo is allowed and varying the weight (P) for dataset 3.

VI. CONCLUSION

In this paper, we used random forests algorithm to improve the accuracy performance of keystroke dynamic biometric authentication. We also proposed a novel algorithm for handling typing mistakes. Simulations results on the publicly available and our collected data sets show that the proposed approach yields much improved performance results. Furthermore, using finger pressure and area along with the timing features significantly improve the performance of the proposed scheme.

Despite the biometric factor, using fixed password strings can be vulnerable to replay attacks. A sophisticated key logger can be implemented to sniff the password strings along with keystroke dynamics, which can then be reused to gain access to the protected system. One approach to mitigate such threat, is to extract the keystroke dynamics from a one-time password (OTP). This approach still poses significant challenges in terms of accuracy. Our future work will consist of investigating such approach and the related challenges.

ACKNOWLEDGMENT

This reasearch is supported by the Jazan University and the Ministry of Education of the Kingdom of Saudi Arabia.

REFERENCES

[1] D. W. Salil Partha Banerjee, "Biometric authentication and identification using keystroke dynamics: A survey," *Journal of Pattern Recognition Research.*, vol. 7, no. 1, 2012.

[2] N. L. Clarke and S. M. Furnell, "Authenticating mobile phone users using keystroke analysis," *Int. J. Inf. Secur.*, vol. 6, no. 1, pp. 1–14, Dec. 2006. [Online]. Available: http://dx.doi.org/10.1007/s10207-006-0006-6

[3] M. Karnan and N. Krishnaraj, "A model to secure mobile devices using keystroke dynamics through soft computing techniques," *International Journal of Soft Computing and Engineering (IJSCE) ISSN*, pp. 2231–2307, 2012.

[4] E. Maiorana, P. Campisi, N. González-Carballo, and A. Neri, "Keystroke dynamics authentication for mobile phones," in *Proceedings of the 2011 ACM Symposium on Applied Computing*, ser. SAC '11. New York, NY, USA: ACM, 2011, pp. 21–26. [Online]. Available: http://doi.acm.org/10.1145/1982185.1982190

[5] M. Karnan and N. Krishnaraj, “Keystroke dynamic approach to secure mobile devices,” in *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, Dec 2010, pp. 1–4.

[6] P. Bours and E. Masoudian, “Applying keystroke dynamics on one-time pin codes,” in *Biometrics and Forensics (IWBF), 2014 International Workshop on*, March 2014, pp. 1–6.

[7] H. Saevanee and P. Bhattarakosol, “Authenticating user using keystroke dynamics and finger pressure,” in *Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE*, Jan 2009, pp. 1–2.

[8] S. Sen and K. Muralidharan, “Putting “pressure” on mobile authentication,” in *Mobile Computing and Ubiquitous Networking (ICMU), 2014 Seventh International Conference on*, 2014, pp. 56–61.

[9] M. Trojahn and F. Ortmeier, “Toward mobile authentication with keystroke dynamics on mobile phones and tablets,” in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, March 2013, pp. 697–702.

[10] B. Draffin, J. Zhu, and J. Y. Zhang, “Keysens: Passive user authentication through micro-behavior modeling of soft keyboard interaction,” in *Mobile Computing, Applications, and Services - 5th International Conference, MobiCASE 2013, Paris, France, November 7-8, 2013, Revised Selected Papers*, 2013, pp. 184–201. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-05452-0_14

[11] “Keystroke dynamics on android platform,” *Procedia Technology*, vol. 19, pp. 820 – 826, 2015. [Online]. Available: http://www.ms.sapientia.ro/~manyi/keystroke.html

[12] K. Killourhy and R. Maxion, “Comparing anomaly-detection algorithms for keystroke dynamics,” in *Dependable Systems Networks, 2009. DSN ’09. IEEE/IFIP International Conference on*, June 2009, pp. 125–134.

[13] M. El-Abed, M. Dafer, and R. El Khayat, “Rhu keystroke: A mobile-based benchmark for keystroke dynamics systems,” in *Proceedings of the 48th IEEE International Carnahan Conference on Security Technology*, 2012.

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The weka data mining software: An update,” *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278