IET Biometrics

Research Article

IET Journals
The Institution of Engineering and Technology

Free-text keystroke dynamics authentication for Arabic language

ISSN 2047-4938
Received on 29th October 2015
Revised on 11th January 2016
Accepted on 8th February 2016
doi: 10.1049/iet-bmt.2015.0101
www.ietdl.org

Arwa Alsultan[1] ✉, *Kevin Warwick*[2], *Hong Wei*[1]

[1]*School of Systems Engineering, University of Reading, Reading RG6 6AH, UK*
[2]*Vice Chancellors Office, Coventry University, Priory Street, Coventry CV1 5FB, UK*
✉ *E-mail: a.f.a.alsultan@pgr.reading.ac.uk*

Abstract: This study introduces an approach for user authentication using free-text keystroke dynamics which incorporates text in Arabic language. The Arabic language has completely different characteristics to those of English. The approach followed in this study involves the use of the keyboard's key-layout. The method extracts timing features from specific key-pairs in the typed text. Decision trees were exploited to classify each of the users' data. In parallel for comparison, support vector machines were also used for classification in association with an ant colony optimisation feature selection technique. The results obtained from this study are encouraging as low false accept rates and false reject rates were achieved in the experimentation phase. This signifies that satisfactory overall system performance was achieved by using the typing attributes in the proposed approach, while typing Arabic text.

1 Introduction

The ongoing quest to find a technique to protect sensitive data and computer systems from harmful imposters, whilst maintaining ease of use, is an important challenge in the field of information security. This paper focuses on a novel method that verifies the identities of users based on their unique typing rhythms in Arabic language. Keystroke dynamics is considered to be an effortless behaviour-based method for user authentication which employs the person's typing patterns for validating his/her identity. As mentioned in [1], keystroke dynamics is 'not what you type, but how you type'. In this approach, the user types in text, as usual, without any extra work to be done for authentication. Moreover, it only involves the user's own keyboard and no other external hardware.

Keystroke dynamics is based on timing features that compute time lapses between two actions on the keyboard such as key press and key release. In this paper, we investigate the use of such timing features with Arabic input. We consider a keyboard-layout-based method to compare timing features of free-text typing samples. A large feature set is deployed in this research for the purpose of trying to find the best representative features of the typing patterns in human behaviour using the least amount of training, specifically while typing in Arabic.

Keystroke dynamics have been studied comprehensively using English input. Other languages have not yet received the same attention as English has in the research literature to date. Languages such as Italian, which share the same alphabet with English, have been included in some research on keystroke dynamics, e.g. [2]. To our knowledge, there has been no reported research that has utilised Arabic input in keystroke dynamics authentication. Use of handwriting identification in Arabic writing has however been reported on in the Arabic-related literature [3]. Therefore, in this paper we are attempting to incorporate Arabic input in keystroke dynamics user authentication. The Arabic language has completely different characteristics to those of English, thus using typing patterns for Arabic input to authenticate users is an important contribution of this paper.

Arabic and English languages are very different to each other. Whereas English is a Germanic language from the Indo-European language family, Arabic is a Semitic language belonging to the Afro-Asiatic language family [4]. Arabic has 28 letters which are completely different from the English alphabet. Moreover, Arabic text is written from right to left which is a unique characteristic for merely Arabic, Urdu and Hebrew scripts [5]. In addition, there is no distinction between upper and lower cases in Arabic. Punctuation rules are much looser than those in English and less commonly used [6].

In this paper, decision trees (DTs) and support vector machines (SVMs) associated with ant colony optimisation (ACO) are used to classify the typing samples collected from participants.

The rest of this paper proceeds as follows. Section 2 briefly introduces keystroke dynamics theory and discusses similar prior research in the area of keystroke dynamics user authentication. Section 3 describes the method developed in this paper, in which we discuss the timing features included in this paper. In Section 4, we present our experimental results and consider the data space under investigation. Discussion about our results and some comparisons with previous studies are also included in this section. The final section concludes the topic and points out our research contributions and future work.

2 Related work

There are two basic classes of keystroke dynamics: namely, fixed-text and free-text [7]. The fixed-text keystroke dynamics method uses the typing pattern of the user while entering a predefined text. This text has been previously used to train the system and is delivered by the user at log-in time. Contrariwise, the free-text keystroke method is considered easier for the user as it overcomes the problem of memorising the text, something that the fixed-text keystrokes method suffers from [8]. As its name suggests in the free-text keystrokes method, the text used for enrolment does not have to be the same as the text used for log-in. Moreover, free-text keystroke dynamics is used for enhancing security through continuous and non-intrusive authentication [9]. Thus, the latter method is the one that has been considered in this paper as it can be applied in many useful settings to aid in real-life situations in addition to the benefit it provides in balancing between security and usability [10].

Keystroke dynamics is utilised in user authentication by extracting timing features at the log-in session and comparing them with the timing features extracted at the enrolment session. These features

IET Biom., 2016, Vol. 5, Iss. 3, pp. 164–169

include, among others: typing latency [11], keystroke duration [1], typing speed and shift key usage patterns [12]. Another feature which requires a specific keyboard for its measurement is typing pressure [13]. If the extracted features are adequately similar, the user is authenticated and if not the user might be denied access or at least asked to provide further identity information.

A large amount of research has been carried out over the years to investigate how keystroke dynamics can aid user authentication. Joyce and Gupta [11] used a statistical method that employs the absolute distances between the means of the signature data and test data; each of which consists of a fixed-text that includes username, password, first name and last name.

Gunetti and Picardi [14] meanwhile introduced an effective method for free-text authentication which was further explored by many other researchers. Their method was based on two measures: a relative (R) measure and an absolute (A) measure. These measures were used to calculate the degree of disorder and the absolute distance between two samples that share some *n*-graphs, i.e. *n*-characters-long letter combinations.

Another technique was introduced by Singh and Arya [15] that considered benefiting from key-pairs for free-text authentication. In that research, a keyboard grouping technique was used for creating timing vectors of flight times entered by the user. The keys were grouped based on their position on the keyboard, which was divided into 8 sections; two left and right halves and then each half divided into 4 lines representing the rows of the keyboard. The Euclidean distance was then used to calculate the similarity between the two vectors.

Other researchers relied on pattern recognition classifying methods such as the work done by Hu *et al.* [16]. They used the *k*-nearest neighbour approach together with the distance measurement proposed by Gunetti and Picardi [14] in order to classify the users' keystroke dynamics profiles. Neural networks have also been used for keystroke pattern classification such as the research conducted by Raghu *et al.* [17], in which they incorporated a three-layered back propagation neural network to verify the identity of users.

It should be indicated that the previously mentioned studies, involved only English input from the user. Whilst such experimentation is very important, there is clearly a lack of language variation used in such systems.

The work done by Gunetti *et al.* [2] is one of the very few research involving languages other than English in keystroke dynamics. In that study, a combination of the two measures developed by Gunetti and Picardi [14] is also used to assess the similarities between the typing patterns using the duration time of the di-graphs in samples typed in both English and Italian. Italian was used since the two languages, i.e. English and Italian, share a considerable number of di-graphs (key-pairs). From experimenting with different combinations of template samples and test samples, the best results were achieved when the user's profile contains samples in both languages yet the performance increases when the language of the testing samples is the dominant language in the samples forming the user profile.

Another study directly relevant to the research reported in this paper was that by Samura and Nishimura [18], in which they conducted a study that examined keystroke dynamics for long free-texts in Japanese language. In this paper, hold time and flight time of Japanese language-specific keystrokes were used as timing features. To compare the test and training timing vectors, a weighted Euclidean distance was used. However, though this paper was applied to Japanese language, the keyboard used was an English standard keyboard. Subjects carried out the typing process by entering the alphabet letters (in English) corresponding to the Japanese characters.

3 Methodology

3.1 Key-pair formation

This research examines the use of Arabic input in the novel approach we introduced in [19] for free-text keystroke dynamics authentication. This approach specifically makes use of the keyboard's key-layout. The technique employs the keystroke features extracted between two keys (key-pair) that are pressed consecutively and have a relationship on the keyboard layout. This relationship depends mainly on the key position of each character on the keyboard in relation to the other characters. Moreover, these relationships can vary depending on the location of the two keys with respect to the overall keyboard layout. The keyboard used in this paper was the standard Arabic keyboard since it is the most commonly used Arabic keyboard [20].

There are five categories for key-pair relationships:

(i) Adjacent: keys located next to each other on the keyboard.
(ii) Second adjacent: keys that are one key apart from each other.
(iii) Third adjacent: keys that are two keys apart.
(iv) Fourth adjacent: keys that are three keys apart.
(v) None adjacent: keys that are more than three keys apart.

Fig. 1 illustrates the key relationship concept when considering the key 'ل'. This is just an example, as the relationship formation can be performed in the same way for all the key-pairs in the typed text.

Each of these relationship categories can fall into one of the following overall locations:

(i) Both keys are on the right-hand side of the keyboard.
(ii) Both keys are on the left-hand side of the keyboard.
(iii) The two keys are located on different sides of the keyboard, i.e. the first key is located on the right-hand side while the second key is on the left or vice versa.

For further explanation, an example is provided in Fig. 2. One colour represents the right-hand side section of the keyboard, whereas the other colour represents the left-hand side. Other characters that are not located in the main part of the keyboard, e.g. the num-pad keys, arrows and the function keys are excluded from the key-pair formation; they are therefore shown in white.

As an example, we demonstrate the key-pairs forming the text 'نتلائم' which is entered from right to left as the following sequence: 'م ئ ا ل ت ن'. The key-pairs are:

- 'ن ت': Adjacent/RightSide.
- 'ت لا': SecondAdjacent/DifferentSide.

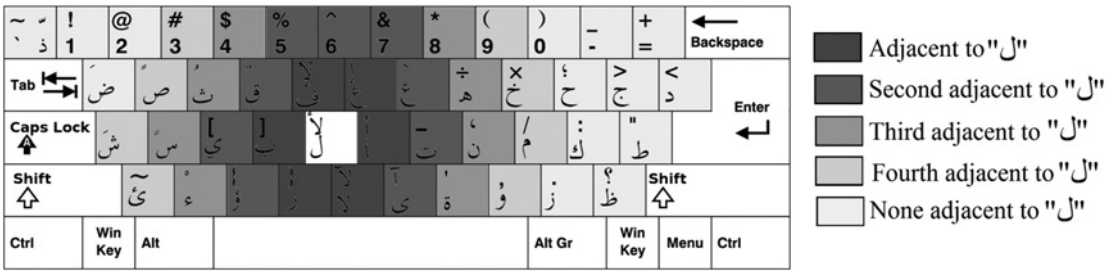

Fig. 1 *Key-pair relationship formation*

Fig. 2 *Overall key location*

- 'لا ى': FourthAdjacent/LeftSide.
- 'ى م': NonAdjacent/DifferentSide.

This process is pursued to break the text down into key-pairs and then classify each key-pair typed on the main part of the keyboard. In total, there are 15 different combinations of key-pairs that any two keys can be classified into.

Given that the key-pairing method significantly boosts the number of key-pairs that can be found and compared in the training and testing samples, it was adopted as a way to increase the soundness of the mean and standard deviation of the timing features. This will help to increase the stability of the timing vectors. This is an obvious benefit of the suggested scheme as it utilises a small amount of typing data in the best possible way. Therefore, it succeeds in authenticating users based on the smallest amount of training possible.

For example, the following training and testing data have only two similar key-pairs ('ال' and 'ير') to which typing times can be compared in the conventional keystroke dynamics authentication process, i.e. without the use of key-pairs [14]. However, this is not the case when using the key-pair method as there are more instances of each key-pair extracted from both the training and testing data.

Training data: 'قرد الغابة صغير'.
Testing data: 'حوت البحر كبير'.

Pairs that involve spaces were discarded because of the work conducted in [15], which provided evidence that a user normally experiences unusual pauses before and after pressing the space key, thereby leading to inconsistent typing behaviour. Moreover, key-pairs that included the backspace key were also excluded for similar reasons.

3.2 Feature definition

Once the key-pairs have been obtained from the users' raw data, the keystroke features are extracted [21]. These features were computed for every key and key-pair using two main values, specifically: the press time (D_n) and the release time (U_n) of each key (n) in milliseconds. In this research, five keystroke features were extracted from each key-pair as shown in Fig. 3:

(i) Dwell time or keystroke duration or hold time: is the time for which a key is pressed until it is released. Consequently, each key-pair has two hold times:
  (a) Hold time for the first key (H1).
  (b) Hold time for the second key (H2).

(ii) Flight time or keystroke latencies: there are three types of latencies:
  (a) Down–Down (DD) or Press–Press time: is the interval time between two successive key presses.
  (b) Up–UP (UU) or Release–Release time: is the interval time between two successive key releases.
  (c) Up–Down (UD) or Release–Press time: is the interval time between a key release and the next key press.

3.3 Feature subset selection and classification

Five timing features were defined for each key-pair appearance in the text. This was done for all 15 types of key-pairs. Therefore, the overall number of timing features was 75 (5 timing features × 15 key-pairs). Table 1 lists all the 75 features extracted from all key-pairs. The feature abbreviations listed in this table combine the key-pair category and the timing feature, for example: 'AR-H1' stands for: Adjacent/RightSide-Hold1 and so on.

Having such a large feature set in its entirety adds more computational cost in addition to raising the complexity of the classification process [22]. Therefore, it is necessary to incorporate a feature subset selection mechanism.

Feature subset selection is considered as an optimisation problem, in which the space of all possible features is scrutinised to recognise the feature or set of features that produce optimal or near-optimal performance, i.e. those that minimise the classification error [7]. ACO proved to be a good candidate for achieving that goal [19].

SVMs have been chosen as a classifier in this research as it is one of the most successful classification techniques [23]. Moreover, for

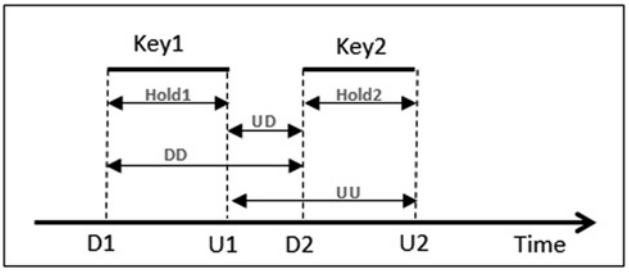

Fig. 3 *Keystroke timing features*

Table 1 Feature set

Key-pair category	Feature set				
Adjacent/RightSide	AR-H1	AR-H2	AR-DD	AR-UU	AR-UD
Adjacent/LeftSide	AL-H1	AL-H2	AL-DD	AL-UU	AL-UD
Adjacent/DifferentSide	AD-H1	AD-H2	AD-DD	AD-UU	AD-UD
SecondAdjacent/RightSide	SR-H1	SR-H2	SR-DD	SR-UU	SR-UD
SecondAdjacent/LeftSide	SL-H1	SL-H2	SL-DD	SL-UU	SL-UD
SecondAdjacent/DifferentSide	SD-H1	SD-H2	SD-DD	SD-UU	SD-UD
ThirdAdjacent/RightSide	TR-H1	TR-H2	TR-DD	TR-UU	TR-UD
ThirdAdjacent/LeftSide	TL-H1	TL-H2	TL-DD	TL-UU	TL-UD
ThirdAdjacent/DifferentSide	TD-H1	TD-H2	TD-DD	TD-UU	TD-UD
FourthAdjacent/RightSide	FR-H1	FR-H2	FR-DD	FR-UU	FR-UD
FourthAdjacent/LeftSide	FL-H1	FL-H2	FL-DD	FL-UU	FL-UD
FourthAdjacent/DifferentSide	FD-H1	FD-H2	FD-DD	FD-UU	FD-UD
NonAdjacent/RightSide	NR-H1	NR-H2	NR-DD	NR-UU	NR-UD
NonAdjacent/LeftSide	NL-H1	NL-H2	NL-DD	NL-UU	NL-UD
NonAdjacent/DifferentSide	ND-H1	ND-H2	ND-DD	ND-UU	ND-UD

IET Biom., 2016, Vol. 5, Iss. 3, pp. 164–169

comparison purposes, DTs were also used in this experiment. DTs were chosen as a rival classifier because the technique follows a completely different mechanism to that of SVMs.

None the less, when using SVMs in classification, feature subset selection was in place in order to reduce redundancy among the features [24]. Contrariwise, feature subset selection was impeded in the building process of the DT where all redundant features were removed [25].

4 Experiment, results and discussion

4.1 Data space

A total of 21 users participated in this paper for data collection. All participants were native Arabic language speakers. They had different levels of typing skills that varied between moderate and very good.

During data collection, the participants were asked to perform two typing tasks. The tasks involved copying text that consisted of around 180 characters. The text employed was an excerpt from an Arabic online newspaper. The text included, in addition to letters, numbers and punctuation marks.

In this research, we used keystrokes produced by typing free-text to authenticate users. Free-text refers to the utilisation of any text for first training and then testing. Importantly the two texts do not have to be the same, as opposed to the use of predefined text in the fixed-text keystroke method, in which the enrolment text and log-in text must be identical [26].

Though the tasks included text that was chosen for the users to type, it is still considered free-text as the text used for training is not related at all to that used for testing [8]. Therefore, based on the definition of free-text [14], all text used in this paper was free-text but with a different method for sourcing the text. In fact, the results produced by the experiments carried out in [27] illustrates that using either un-copied or copied text has no effect on the results of free-text keystroke systems. Copied text was however provided for the participants to ease the process of data collection.

Users were directed to enter the samples in the most natural way possible, i.e. the same way they usually follow when typing. They were also allowed to enter carriage returns and backspaces if needed. Data collection was performed by a graphical user interface (GUI) programme implemented using the C++ language. The application was downloaded on the users' personal machines to maximise their comfort, on the basis that they are more familiar with their own machine and its surroundings. Therefore, they were able to feel more at ease, and thus, to perform the typing tasks in a manner closer to that of their real typing behaviour. Thus, an uncontrolled environment was adapted due to the fact that the data was collected wherever and whenever it was convenient to the user, thereby as much as possible providing a realistic representation of the normal conditions for the user.

Moreover, on observing the data collected in this experiment, a number of outliers were detected. Outlier data was identified to be as much as three standard deviations above or below the mean, as was suggested in [11]. These particularly very large or very small data points were discarded from the final data as it was considered that they represented noise that might affect the overall system performance.

In addition, it was seen as preferable to normalise the data before handing it to the machine learning technology [28]. Therefore, all the data was normalised to be between [0,1] to add a sense of uniformity to the data as otherwise attributes in greater numeric ranges might have dominated those in smaller numeric ranges [29].

The final step of data pre-processing involved creating the timing vector and storing it in the database as the user's profile. This process was carried out by combining the text from the two tasks and, then, dividing it into eight equal parts. Each one of the eight parts was considered as a single typing sample, the features from which were extracted and its mean calculated and stored separately. Therefore, eight samples per subject were included in the analysis phase for classifier training and testing.

Though there were 15 key-pairs, from which 75 timing features were captured, there were not enough instances that appeared in the used text for some of the key-pairs which made it unfeasible to include them in the final feature set, the omitted key-pairs were: NonAdjacent/RightSide, ThirdAdjacent/LeftSide, FourthAdjacent/LeftSide and NonAdjacent/LeftSide. In total 20 timing features were excluded from the final feature set; resulting in the inclusion of only 55 features in the final feature vector.

4.2 Experiment and results

After creating users' profiles, feature subset selection was performed using ACO [30] for each user's data. The selected features were then passed to the SVMs machine learning mechanism in order to be used as the basic data for differentiating between classes. The radial basis kernel multiclass SVMs classification process [31] was implemented on MATLAB with the aid of the LIBSVM library [32].

The DT technique, on the other hand, is capable of performing feature selection in the tree building phase [25]. Therefore, this was fed with the complete set of features. The Statistics toolbox in MATLAB was used to fit the tree and predict the class of each of the test data.

Classification, for both classifiers, was carried out through cross-validation which is a statistical sampling technique that aims to ensure that every example from the original dataset has the same chance of appearing in the training and testing set. We followed the leave-one-out cross-validation protocol which is a special case of the well-known n-fold cross-validation [33].

N-fold cross-validation divides the data up into n chunks and trains n times, treating a different chunk as the test sample each time; such that for each of n experiments, it uses $n-1$ folds for training and the remaining one for testing. Leave-one-out cross-validation is exactly the same except that all chunks contain only a single sample.

In our experiment, eight samples were used to perform eight cross-validation experiments. Seven of the samples were treated as the training sample set and the remaining sample was regarded as the testing sample. In each experiment, a different sample was selected to act as the test data.

Furthermore, two error rates were used to infer the performance: namely, false accept rate (FAR) and false reject rate (FRR). FAR is the percentage of impostors who have successfully gained access to the system, whereas FRR indicates the percentage of legitimate users who were denied access to the system [7]. In both cases, it is therefore desirable for these figures to be as low as possible.

SVMs have a slight performance advantage over DTs, as shown in Table 2. This is due to the fact that SVMs are more advanced in distinguishing between classes in similar situations [34]. The classification is done in SVMs by performing optimisation to find the separating hyperplane [35], whereas classification in DTs is purely based on rules [36]. It is worth noting that by using a more sophisticated subset selection mechanism such as ACO this has also contributed to the SVM's superior results [37]. The subset selection in DTs, on the other hand, is relatively primitive compared with ACO as it is carried out internally in the tree building stage [25].

The features selected by the ACO were AR-H2, TD-H1, TD-H2, TD-DD and FD-H2. Meanwhile, the features selected by DTs were: TR-UD, TD-H1, TD-DD and FD-DD. The two subsets are different except for two features: TD-H1and TD-DD. It is noted that the features selected by DTs are dominated by latency features whilst the features selected by ACO are dominated by duration features.

Table 2 System performance using Arabic input

	FAR	FRR
DT	0.205	0.512
SVMs	0.169	0.423

Table 3 System performance using English input

	FAR	FRR
DT	0.281	0.702
SVMs	0.245	0.613

This partially explains the lower error rate produced by SVMs classification. As found in [38], duration time appears to be a more reliable method to capture a user typing pattern compared with latency in systems concerned with user authentication.

4.3 Discussion and comparison of Arabic and English

In this paper, we performed the same experiment using English input on the same 21 subjects. The results of the English input experiment are shown in Table 3. Similar to Arabic input, SVMs proved to outperform DTs as they produced lower error rates. SVMs used features selected by ACO in the classification stage, these features were: AL-H1, AL-H2, AD-UU, AD-UD and SR-H2. Meanwhile, with DTs subset selection was performed in the tree building stage. The features chosen by DTs in the classification were: SL-UU, TL-DD, FL-UD, FD-DD and ND-DD. The features subset selected by the ACO consisted of both duration times and latency times, while the features selected by DTs were exclusively latency times. As with Arabic input, this contributed to the superiority of the SVMs system performance as duration time has been shown to produce better system performance [38].

It is worth mentioning that we noted a difference between the key-pairs in English and Arabic. In Arabic the number of key-pairs on the right-hand side of the keyboard is more than those on the left as most of the commonly used letters are on the right-hand side of the keyboard. English, on the other hand, has a greater number of the most used letters (letters such as *e*, *t*, *a*, *s* and *r*) on the left-hand side of the keyboard, thus key-pairs from that side are larger in number than those on the right-hand side. Owing to that, the key-pairs that were used to create the users' Arabic profile have some differences to those used for creating the user's English profile, as mentioned in Section 4.1. The key-pairs included in the English experiment can be found in previous research [19].

Moreover, Arabic input results were generally better than those based on English input due to the fact that all of the subjects chosen to be part of the experiments of this paper were native Arabic speakers and ten of the 21 subjects do not type in English regularly. As Arabic speakers are used to typing in Arabic, their typing skills have developed in Arabic and they are more familiar with an Arabic keyboard and how to operate it. This provides their typing with enough uniformity to be used to correctly identify the users based on their typing patterns. In addition, subjects who are not familiar with English typing have less familiarity with an English keyboard and typing in that language. Therefore, the absence of English experience has caused non-English natives to lack the typing consistency needed to identify users based on their typing patterns [39].

The same key-pairing approach has been used in our previous research for user authentication using English text [19], in which the SVMs/ACO method resulted in an FAR of 0.001 and an FRR of 0.504. The performance of that experiment outperformed the English experiment we performed in this paper for two reasons. First, the features selected in that experiment were: SR-H2, FL-H2, FD-H2, ND-H1 and ND-UD. Thus, the selected feature subset is dominated by duration times; this contributed to the superiority of the previous study system performance as duration time has better distinguishing capabilities in keystroke dynamics [38]. Moreover, nine of the 21 participants in the previous English experiment were native English speakers and the rest were very familiar with English and were used to typing in English on a daily basis. This clearly improves the consistency of the typing patterns of such users compared with the subjects in this new study, in which the majority were not familiar with English typing.

Both experiments provided results demonstrating the effect that the most commonly used language has on system performance. Lower error rates are achieved when the system uses the native language for most of the users or a language that most of the users are familiar with. In the experiment performed in this research, the use of Arabic language was shown to achieve higher performance as all of the subjects involved in the experiment were either native to Arabic or more familiar with Arabic. In contrast, in the previous research, the English experiment yielded good results due to the participants being either native to English or more familiar with English typing.

To compare the results we found from Arabic native speakers with results from English native speakers, we conducted a further experiment. In this test, we collected English data from eight English natives and analysed it in a manner similar to that used to analyse the Arabic data. Employing SVMs/ACO, system performances of 0.089 FAR and 0.125 FRR were achieved using English input. This result is considered satisfactory and it demonstrates that using the native language of the users affects the system performance positively. This agrees with the conclusion we made using Arabic native speakers' data. Yet more investigation is needed to achieve better understanding of the English/Arabic native speakers' typing differences and similarities.

5 Conclusions

In this paper, we examined the usefulness of applying free-text keystroke dynamics user authentication on Arabic text by an original keyboard's key-layout-based method. This key-pairing approach works by classifying every two characters typed consecutively based on their relation to each other and their overall location on the keyboard. For each key-pair, five timing features were extracted to be used in the user's feature vector.

SVMs and DTs were employed to classify individuals based on the proposed timing features. The experiment produced good results considering the fact that it used free-text for user authentication. Moreover, it accomplished user authentication based on the smallest amount of training possible. The FAR and FRR rates were both satisfactory with the FAR being the slightly better of the two.

This paper proved that the proposed method has been successful in authenticating users based on their Arabic typing. The method was originally created to be used with English typing, yet it has crossed over to Arabic input successfully. Moreover, in the comparative study SVMs produced lower error rates compared with DTs. Duration times also proved to contribute more in increasing the system performance when compared with the latency times.

In addition, experimenting with two languages showed that the user's familiarity with a certain language has a high impact on the user's typing patterns in that language. This considerably affects the system performance as lower error rates are produced from systems incorporating a language native or familiar to the users.

There is much more that can be done to improve this approach, one example of which is to expand the typing features to include other non-timing features such as typing speed, error rate and special keys usage patterns. Experimenting with different classification methods might also contribute positively to the overall system performance. Moreover, experimenting with other languages, which have a different alphabet to English such as Chinese or Thai can be carried out to understand how they compare with English and Arabic.

6 References

1 Monrose, F., Rubin, A.: 'Authentication via keystroke dynamics'. Proc. Fourth ACM Conf. on Computer and Communications Security, 1997, pp. 48–56

2 Gunetti, D., Picardi, C., Ruffo, G.: 'Keystroke analysis of different languages : a case study'. Proc. Advances in Intelligent Data Analysis 2005, vol. VI, no. 2, pp. 133–144

3 Awaida, S.M., Mahmoud, S.A.: 'State of the art in off-line writer identification of handwritten text and survey of writer identification of Arabic text', *Educ. Res. Rev.*, 2012, **7**, (20), pp. 445–463
4 Katzner, K.: 'The languages of the world' (Routledge Ltd., London, 1975)
5 Cooper, R.L., Olshtain, E., Tucker, G.R., *et al.*: 'The acquisition of complex English structures by adult native speakers of Arabic and Hebrew', *Lang. Learn.*, 1979, **29**, (2), pp. 255–275
6 Versteegh, K.: 'The Arabic language' (Edinburgh University Press, Edinburgh, 2014, 2nd edn.)
7 Karnan, M., Akila, M., Krishnaraj, N.: 'Biometric personal authentication using keystroke dynamics: a review', *Appl. Soft Comput.*, 2011, **11**, (2), pp. 1565–1573
8 Banerjee, S.P., Woodard, D.L.: 'Biometric authentication and identification using keystroke dynamics: a survey', *J. Pattern Recognit. Res.*, 2012, **7**, pp. 116–139
9 Bours, P., Mondal, S.: 'Performance evaluation of continuous authentication systems performance evaluation of continuous authentication systems', *IET Biometrics*, 2015, **4**, (4), pp. 1–7
10 Alsultan, A., Warwick, K.: 'Keystroke dynamics authentication: a survey of free-text methods', *Int. J. Comput. Sci. Issues*, 2013, **10**, (4), pp. 1–10
11 Joyce, R., Gupta, G.: 'Identity authentication based on keystroke latencies', *Commun. ACM*, 1990, **33**, (2), pp. 168–176
12 Lau, E., Liu, X., Xiao, C., *et al.*: 'Enhanced user authentication through keystroke biometrics'. Proc. Computer and Network Security, 2004
13 Lv, H.R., Lin, Z.L., Yin, W.J., *et al.*: 'Emotion recognition based on pressure sensor keyboards'. Proc. IEEE Int. Conf. Multimedia and Expo, 2008, pp. 1089–1092
14 Gunetti, D., Picardi, C.: 'Keystroke analysis of free text', *ACM Trans. Inf. Syst. Sec.*, 2005, **8**, (3), pp. 312–347
15 Singh, S., Arya, K.V.: 'Key classification: a new approach in free text keystroke authentication system'. Proc. Third Pacific-Asia Conf. on Circuits, Communications and System, 2011, pp. 1–5
16 Hu, J., Gingrich, D., Sentosa, A.: 'A K-nearest neighbor approach for user authentication through biometric keystroke dynamics'. Proc. IEEE Int. Conf. on Communications, 2008, pp. 1556–1560
17 Raghu, D., Jacob, C.R., Bhavani, Y.V.K.D.: 'Neural network based authentication and verification for web based key stroke dynamics', *Int. J. Comput. Sci. Inf. Technol.*, 2011, **2**, (6), pp. 2765–2772
18 Samura, T., Nishimura, H.: 'Keystroke timing analysis for individual identification in Japanese free text typing'. Proc. ICCAS-SICE, 2009, pp. 3166–3170
19 Alsultan, A., Warwick, K., Wei, H.: 'Feature subset selection for free-text keystroke dynamics' (John Wiley & Sons, NJ, USA, 2015), in press
20 Malas, T.M., Taifour, S.S., Abandah, G.A.: 'Toward optimal Arabic keyboard layout using genetic algorithm'. Proc. Ninth Int. Middle Eastern Multiconference on Simulation and Modeling (MESM 2008), 2008, pp. 50–54
21 Teh, P.S., Teoh, A.B.J., Ong, T.S., *et al.*: 'Statistical fusion approach on keystroke dynamics'. Proc. Third Int. IEEE Conf. on Signal-Image Technologies and Internet-Based System, 2007, pp. 918–923
22 Saeys, Y., Inza, I., Larrañaga, P.: 'A review of feature selection techniques in bioinformatics', *Bioinformatics*, 2007, **23**, (19), pp. 2507–2517
23 Burges, C.J.C.: 'A tutorial on support vector machines for pattern recognition', *Data Min. Knowl. Discov.*, 1998, **2**, (2), pp. 121–167
24 Shanmugapriya, D., Padmavathi, G.: 'An efficient feature selection technique for user authentication using keystroke dynamics', *Int. J. Comput. Sci. Netw. Secur.*, 2011, **11**, (10), pp. 191–195
25 Russell, S., Norvig, P.: 'Artificial intelligence: a modern approach' (Prentice-Hall, Englewood Cliffs, 2010, 3rd edn.)
26 Teh, P.S., Teoh, A.J., Yue, S.: 'A survey of keystroke dynamics biometrics', *Sci. World J.*, 2013, pp. 1–24
27 Killourhy, K.S., Maxion, R.A.: 'Free vs. transcribed text for keystroke-dynamics evaluations'. Proc. ACM Int. Conf. Proceeding Series, 2012, pp. 1–8
28 Hsu, C.W., Chang, C.C., Lin, C.J.: 'A practical guide to support vector classification'. Technical Report, Department of Computer Science, National Taiwan University, 2003
29 Kantardzic, M.: 'Data mining: concepts, models, methods, and algorithms' (2011, 2nd edn.)
30 Jensen, R.: 'Performing feature selection with ACO', *Swarm Intell. Data Min.*, 2006, **34**, pp. 45–73
31 Ambwani, T.: 'Multi class support vector machine implementation to intrusion detection'. Proc. the Int. Joint Conf. on Neural Networks, 2003, vol. 3, pp. 2300–2305
32 Chang, C.C., Lin, C.J.: 'LIBSVM: a library for support vector machines'. 2001, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
33 Refaeilzadeh, P., Tang, L., Liu, H.: 'Cross-validation', *Encycl. Database Syst.*, 2009, pp. 532–538
34 Caruana, R., Niculescu-Mizil, A.: 'An empirical comparison of supervised learning algorithms'. Proc. 23th Int. Conf. on Machine Learning, 2006, pp. 161–168
35 Fletcher, T.: 'Support Vector Machines Explained', pp. 1–19, 2009. Available at http://www.sutikno.blog undip.ac.id/files/2011/11/SVM-Explained. pdf., accessed 06 June 2013
36 Safavian, S.R., Landgrebe, D.: 'A survey of decision tree classifier methodology', *IEEE Trans. Syst. Man Cybern.*, 1991, **21**, (3), pp. 660–674
37 Dorigo, M., Stützle, T.: 'Ant colony optimization' (Massachusetts Institute of Technology, Cambridge, Massachusetts, 2004)
38 Robinson, J.A., Liang, V.M., Chambers, J.A.M., *et al.*: 'Computer user verification using login string keystroke dynamics', *IEEE Trans. Syst. Man Cybern. A, Syst. Humans*, 1998, **28**, (2), pp. 236–241
39 Thorson, H.: 'Using the computer to compare foreign and native language writing processes: a statistical and case study approach', *Mod. Lang. J.*, 2000, **84**, (2), pp. 155–169