Improving the Performance of Free-text Keystroke Dynamics Authentication by Fusion

Alsultan, A, Warwick, K & Wei, H

Author post-print (accepted) deposited by Coventry University's Repository

Original citation & hyperlink:
Alsultan, A, Warwick, K & Wei, H 2017, 'Improving the Performance of Free-text Keystroke Dynamics Authentication by Fusion' *Applied Soft Computing*, vol (in press), pp. (in press)
https://dx.doi.org/10.1016/j.asoc.2017.11.018

DOI 10.1016/j.asoc.2017.11.018
ISSN 1568-4946
ESSN 1872-9681

Publisher: Elsevier

NOTICE: this is the author's version of a work that was accepted for publication in *Applied Soft Computing*. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in *Applied Soft Computing*, [(in press), (2017)] DOI: 10.1016/j.asoc.2017.11.018

© 2017, Elsevier. Licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International
http://creativecommons.org/licenses/by-nc-nd/4.0/

Copyright © and Moral Rights are retained by the author(s) and/ or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This item cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder(s). The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.

This document is the author's post-print version, incorporating any revisions agreed during the peer-review process. Some differences between the published version and this version may remain and you are advised to consult the published version if you wish to cite from it.

Improving the Performance of Free-text Keystroke Dynamics Authentication by Fusion

Arwa Alsultan [a*], Kevin Warwick [b], Hong Wei [c]

[a] Information Technology Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

[b] Vice Chancellors Office, Coventry University, Priory Street, Coventry CV1 5FB, UK

[c] Computer Science Department, School of Mathematical, Physical and Computational Sciences, University of Reading, Reading RG6 6AH, UK

[*] Corresponding author: afalsultan@ksu.edu.sa

Abstract

Free-text keystroke dynamics is invariably hampered by the huge amount of data needed to train the system. This problem has been addressed in this paper by suggesting a system that combines two methods, both of which provide a reduced training requirement for user authentication using free-text keystrokes. The two methods were fused to achieve error rates lower than those produced by each method separately. Two fusion schemes, namely: decision-level fusion and feature-level fusion, were applied. Feature-level fusion was done by concatenating two sets of features before the learning stage. The two sets of features were: a timing feature set and a non-conventional feature set. Moreover, decision-level fusion was used to merge the output of two methods using majority voting. One is Support Vector Machines (SVMs) together with Ant Colony Optimization (ACO) feature selection and the other is decision trees (DTs). Even though the classifiers using the parameters merged at feature level produced low error rates, its results were outperformed by the results achieved by the decision-level fusion scheme. Decision-level fusion was employed to achieve the best performance of 0.00% False Accept Rate (FAR) and 0.00% False Reject Rate (FRR).

Keywords Free-text keystroke dynamics authentication, feature-level fusion, decision-level fusion, SVMs, ACO, decision tree

1 Introduction

Keystroke dynamics is an effortless behaviour-based method for authenticating users, which employs the person's typing patterns for validating his/her identity. As mentioned by [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. These criteria make keystroke dynamics an excellent alternative or add on to the conventional ID/password authentication scheme.

Unfortunately, passwords are prone to social engineering and can be easily cracked using methods such as dictionary attack and brute force attack. Therefore, users are obliged to use extreme measures to safeguard their passwords, a procedure which includes remembering long and complex passwords in addition to the need for changing their passwords periodically [2]. This causes frustration and apprehension for users, especially when a single user is most likely responsible for more than a hand-full of ID/passwords spread over multiple systems.

However, the main drawback of keystroke dynamics authentication is the large amount of training data it requires. Typing large amounts of text in the enrolment phase is time consuming and not user-friendly. A key-pairing method, which is based on the keyboard's key-layout, has been suggested as a way to enable one user's typing pattern to be distinguished from another user's. The method extracts several timing features from specific key-pairs. This technique was developed to use the smallest amount of training data in the best way possible. In addition, non-conventional features were also defined and extracted from the input stream typed by the user in order to understand typing behaviours based on limited input data.

As fusion was proven to reduce the error rate in classification tasks compared with single classifiers [3], these two techniques were fused in order to increase the performance of keystroke recognition whilst using a small amount of training data. In this study, we apply two different types of fusion techniques, namely: feature-level fusion and decision-level fusion. Specifically, this work attempts to implement both kinds of fusion and then compare between the two methods in order to find the fusion technique that produces the best recognition rate in free-text keystroke dynamics systems with limited training.

The feature-level fusion is done by joining keystroke timing features and non-conventional typing features before the learning phase. Meanwhile the decision-level fusion is done by combining the output of a method involving timing features and SVMs/ACO and another method utilizing non-conventional features and decision trees. Both SVMs and DTs are classifiers that follow non-iterative approaches.

The rest of this paper proceeds as follows. Section 2 introduces keystroke dynamics theory and describes some of the work previously carried out in the area of keystroke dynamics user authentication. Section 3 discusses the feature sets used in this experiment. Section 4 describes the different fusion techniques. In Section 5, we point to the experimental results and discussion, in which the data space and the experimental results are indicated. A discussion about our results and some comparisons with previous studies is also performed in this section. The final section concludes the topic and points out our research contributions and future work.

2 Keystroke Dynamics

There are two basic classes of keystroke dynamics, namely: fixed-text and free-text [4]. 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 memorizing the text, something that fixed-text keystrokes suffers from. As its name suggest in free-text keystrokes, 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 nonintrusive authentication [5]. This is done by authenticating users based on freely typed keystrokes [6]. 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 [4]. Nonetheless, long text is provided by the user to train the system at the enrolment phase [5].

Keystroke dynamics is utilized in user's authentication by extracting timing features at the log-in session and comparing them with the timing features extracted at the enrolment session. These features include, among others: typing latency [7], keystroke duration [1], typing speed and shift key usage patterns [8]. Another feature which requires a specific keyboard for its measurement is typing pressure [9]. 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 in user authentication. Joyce and Gupta [7] 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.

Moreover, Gunetti and Picardi [10] introduced an effective method for free-text authentication which was further explored by many other researchers. Their method was based on two measures: relative (R) measure and 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.

Other researchers relied on pattern recognition classifying methods such as the work done by Hu et al. [11]. They used the k-nearest neighbor approach together with the distance measurement proposed by Gunetti and Picardi [10] 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. [12] in which they incorporated a three-layered back propagation neural network to verify the identity of users.

Furthermore, a research that has considered fusion in keystroke dynamics is that conducted by Teh et al. [13]. The authors of this research proposed a fusion between two methods. The first being the Gaussian similarity score between a reference template and a test data template. The second being the Direction Similarity Measure (DSM) for comparing the typing patterns of the user. The two scores were fused by using a weighted sum rule. Fifty participants were requested to type-in their username, password and a special fixed phrase repeatedly ten times. The performance achieved using only the Gaussian probability density function yielded an EER of 11.6897%, while the performance of using only the Direction Similarity Measure produced an EER of 19.74%. Combining the two methods delivered the best result of an EER of 6.36%.

Hocquet et al. [14] performed a study for authenticating users using a fusion of three methods in. The first used the mean and the variance of each latency time and compared it to a threshold. The second method used a measure of typing rhythm disorder where the time was classified into five different classes according to the speed. The difference between the numbers of the classes in the profile data and in the test data was calculated and then the sum of all these differences was compared to a threshold. The third and last method was based on the ranks of the latencies; this was performed by ordering the latencies based on their speed. The latency time of each observation was ordered from the slowest to the fastest for each user's profile. The Euclidean distance between the user's profile and the new data was then used to guess if the new observation belonged to that user. Even though these methods work well on their own, the authors studied the possibility of combining all three with the help of a fusion rule after normalizing the

scores from each method. For testing these methods, 15 users were asked to give a login password ten times, followed by 30-to-100 log-in attempts over a six months period. The results of using each of these techniques separately gave a best performance of 3.70% EER. The fusion of all three methods, on the other hand, significantly improved the performance to an EER of around 1.8%.

Moreover, Curtin et al. [15] employed feature fusion, in which the authors combined non-conventional features and traditional keystroke dynamics in user authentication. A total of 58 features were used. In addition to the usual key-press duration and di-graph's down-down and down-up features, other features were utilized in this study; such as: typing speed and the percentages of key presses of some special characters including punctuation and space bar. Other features captured the editing pattern of the user which includes the usage of specific keys i.e. Home, End, Arrow keys, Backspace, Delete keys, Insert, shortcut keys … etc. A Nearest Neighbour classifier which uses Euclidean distance was used in this study to compare the feature vector of the test sample and training set. An accuracy of 0.985 was produced by a dataset consisting of eight individuals.

Based on the literature, not many studies have shown an interest in understanding the differences between using decision-level and feature-level fusion in the area of free-text keystroke dynamics authentication. In this study we therefore explore two fusion techniques with the aim of finding which one is more suitable for the task of uniquely recognizing the typist involved using reduced training.

3 Keystroke Features

Timing features and non-conventional features are utilized in this study. A description of both features is presented in this section

3.1 Timing Features

The timing features used in this study are 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.

There are five categories for key-pair relationships:

1. Adjacent: keys located next to each other on the keyboard.

2. Second adjacent: keys that are one key apart from each other.
3. Third adjacent: keys that are two keys apart.
4. Fourth adjacent: keys that are three keys apart.
5. None adjacent: keys that are more than three keys apart.

An example is provided in Fig. 1. demonstrating the key relationship concept; while considering the key 'G'.

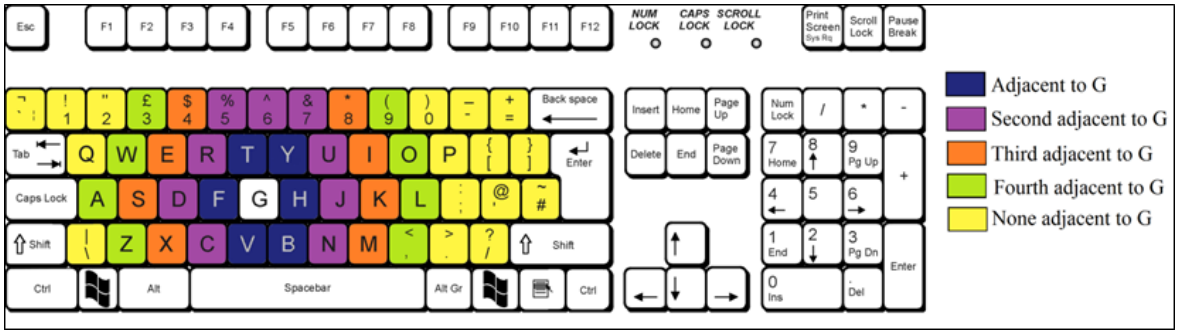

Fig. 1: Key-pair classification.

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

1. Both keys are on the right hand side of the keyboard.
2. Both keys are on the left hand side of the keyboard.
3. 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.

The text is broken down into di-graphs, or key-pairs, and each di-graph typed on the main part of the keyboard is classified using the above categories and locations. In total, there are fifteen different combinations of key-pairs that any two keys can be classified into. Based on the previously explained technique, the key-pair "vr" is categorized as: SecondAdjacent/LeftSide.

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 adapted as a way to increase the soundness of the means of the timing features extracted from these key-pairs. This will help to increase the stability of the timing vectors. This is a clear benefit of the suggested scheme because it enables it to facilitate a small amount of text to compare two samples, i.e. uses a small amount of typing data in the best possible way.

For example, given the following training data: “University of Reading” and testing data: “Systems Engineering,” there are only two similar key-pairs (“in” and “ng”) whose typing times can be compared in the authentication process, using standard keystroke schemes such as the one introduced in [10]. However, this is not the case when using the key-pairing method introduced here because there are more instants for each kind of key-pair extracted from both the training and testing data.

Once the key-pairs have been obtained from the users’ raw data, the keystroke features are extracted [13]. 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. These features are (as shown in Fig. 2):

1. Hold time: is the time a key is pressed until it is released. Consequently, each key-pair has two hold times:
    a. Hold time for the first key (H_1).
    b. Hold time for the second key (H_2).

2. Keystroke latencies: are features that involve computing the time elapses between two actions that were performed on two key. There are three types of latencies:
    a. Down-Down (DD): is the interval time between two successive key presses.
    b. Up-UP (UU): is the interval time between two successive key releases.
    c. Up-Down (UD): is the interval time between a key release and the next key press.

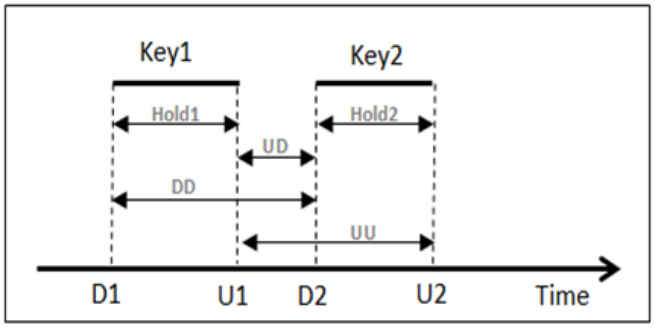

Fig. 2: Timing features for a key-pair.

Thus, five timing features were defined for each key-pair appearance in the text. This was done for all fifteen 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 features abbreviations listed in

the table combine the key-pair category and the timing feature, for example: "AR-H1" stands for: Adjacent/RightSide-Hold1 and so on.

Table 1: Overview of the timing features.

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

The timing vector (V) for a single sample that is stored it in the database of a user consist of eleven sub-vectors representing each key-pair. Thus, the following equations show a timing vector that describes a user's single typing sample:

$$V = \{V_{AR}, V_{AL}, V_{SR}, V_{SL}, V_{SD}, V_{TR}, V_{TL}, V_{TD}, V_{FL}, V_{FD}, V_{ND}\} \quad (1)$$

$$V_{AR} = \{\mu_{AR\text{-}H1}, \mu_{AR\text{-}H2}, \mu_{AR\text{-}DD}, \mu_{AR\text{-}UU}, \mu_{AR\text{-}UD}\} \quad (2)$$

⋮

$$V_{ND} = \{\mu_{ND\text{-}H1}, \mu_{ND\text{-}H2}, \mu_{ND\text{-}DD}, \mu_{ND\text{-}UU}, \mu_{ND\text{-}UD}\} \quad (3)$$

Where:

$\mu_{AR\text{-}H1}$: denotes the mean of the H_1 timing feature for all adjacent key-pairs on the right side of the keyboard in that sample.

$\mu_{AR\text{-}H2}$: denotes the mean of the H_2 timing feature for all adjacent key-pairs on the right side of the keyboard in that sample.

… and so on.

Thus, the data base consisted of twenty-five users' profiles. Each user's profile included eight vectors similar to the one in Equation 1 representing each of the eight samples for each individual.

The key-pairing method introduced in this paper is a new method which deals with key-pairs to extract specific timing features from the typed text as appose to standard keystroke dynamics research such as the ones in [9, 15, 16]. Standard keystroke dynamics research involves comparing two samples based on the timing features of the di-graphs shared between samples. These di-graphs are specific two characters typed after each other with no concern about the key-pair group they belong to, i.e. example of some di-graphs: "em", "et", "as"… etc. Nonetheless, using such di-graphs in free-text input is challenged by the need to collect the same di-graphs from training and testing samples to carry-out the comparison task. Therefore, it is hampered by using a large amount of text.

3.2 Non-conventional Features

Non-conventional typing features are extracted collectively during the whole text input, in which more information is available, such as the percentage of performing a specific action on the keyboard. This enlarges the amount of information that can be extracted from user input and therefore assembles better indications about his or her typing behaviour using a small amount of input data.

The non-conventional features used here include two types of typing features, namely: semi-timing features and editing features. Both categories are explained in this section.

3.2.1 Semi-timing Features

Semi-timing features are different from the standard timing features, as the time calculation followed in this category is slightly different from that of the regular timing features. These features have a collective property, as all of them are calculated during the whole typing time. The features included in this category are:

1. Word-per-Minute (WPM): measures the user's average typing speed.

2. Negative Up-Down (negUD): measures the percentage of negative Up-Down actions detected in the user's typing stream. Negative Up-Down is due to the overlap happening between two successive keys being typed. This particular typing behaviour is found in the typing stream of users who have the tendency to press the second key before releasing the first one. Fig. 3. illustrates a negative UD caused by two-key sequences overlapping. It has been found in the experimentation undertaken in this study that some users have absolutely no negative UDs whilst others have a fair amount, which was consistent in all the typing tasks they produced. This gives a good indication that comparing the percentage of negative UDs can be a good method to assist in user recognition.

3. Negative Up-Up (negUU): measures the percentage of negative Up-Up actions detected in the user's typing stream. Negative Up-Up occurs when the typist tends to release the second key before releasing the first key. Negative UU only happens when there is a negative UD between the two successive keys. Moreover, this characteristic happened with a few of the volunteers who participated in the data collection and it was consistent among all of their typing tasks.

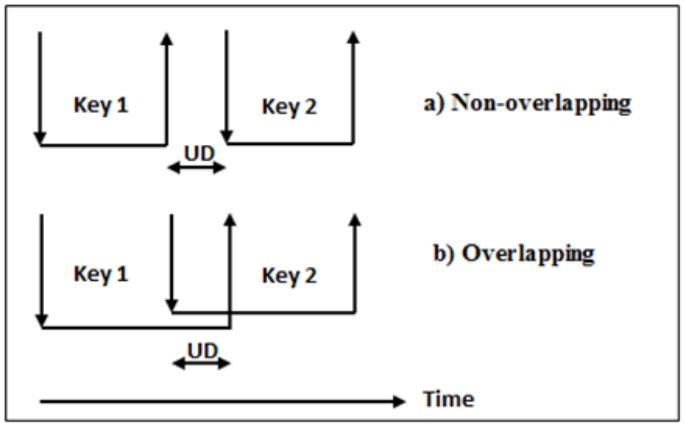

Fig. 3: Negative UD caused by overlapping keystroke events.

3.2.2 Editing Features

Editing features does not give any attention to the time a user spends typing, rather it considers the way a user goes about the process of typing. The features included in this category are:

1. Error rate: captures the percentage of times a user performs a typing error.

2. Capital letters incorporation practices: this subcategory is concerned with the specific ways a capital letter is included in the user's typing stream. This is usually done using either Caps Lock key or shift key. It has been noted that if a user has the habit of using the CapsLock key, then he or she will hardly ever use

the shift key for capitalizing letters, and vice versa. Therefore, using these two attributes simultaneously might lead to better understanding of the user's editing habits. Thus, the following features are employed:

a. CapsLock usage: calculates the percentage of the CapsLock keys being used to produce capital letters in a given typing task.

b. Shift Key usage: this subcategory has two different aspects of user's habits. The first shift key usage attribute is the right/left shift key choice. Some users use strictly the right shift or strictly the left shift and others alternate between the two [8]. The second attribute is the order of which the shift/letter keys are released. The shift key is always pressed before the letter key if the user is intending to produce a capital version of that letter. However, there are two orders that users go about when releasing those keys, they either release the letter key before releasing the shift key or they release the letter key after releasing the shift key. Based on the previous observations, four different features that combine the two aspects of shift key usage are used:

i. Percentage of Right Shift released after letter (RSA).
ii. Percentage of Right Shift released before letter (RSB).
iii. Percentage of Left Shift released after letter (LSA).
iv. Percentage of Left Shift released before letter (LSB).

Table 2 gives an overview of all the nine non-conventional typing features used.

Table 2: Overview of the non-conventional typing features.

Category	***Features***
Semi-Timing Features	WPM
	negUD
	negUU
Editing Features	Error Rate
	CapsLock Usage
	RSA
	RSB
	LSA
	LSB

4 Fusion

Decision support systems (DSS) are schemes to create a model that is able to produce correct decisions given a minimum amount of input data. There are two different ways to go about DSS [18]. The first of

which suggests that the progress of DSS should be based on continuous improvement of existing methods and establishing new ones. The second approach recommends combining existing methods that perform well, anticipating that better results will be achieved as the limits of the existing individual method are reached and it is not possible to develop anything better. Such fusion of information seems to be worth applying in terms of uncertainty reduction. As each of the individual methods produce some errors, different methods performing on different data should produce different errors. Assuming that all individual methods perform well, a combination of such methods should reduce the overall classification error [18].

Fusion of data/information can be carried out on three levels: data-level fusion, feature-level fusion, and decision-level fusion (also called classifier-level fusion) [19]. Data-level fusion combines multiple sensor data that measures correlated parameters [20]. Using multiple sensor systems has many advantages including: higher signal-to-noise ratio, increased robustness and reliability in the case of sensor failure, reduced uncertainty and increased confidence. Thus, integration of data from a multiple sensor system, i.e. data-level fusion, is important to improve decision making [20].

In feature-level fusion, feature sets obtained from several data sources can be fused to create a new feature set representing a single object. The most common way to accomplish feature level fusion is by a simple concatenation of the feature sets obtained from multiple information sources [21]. The concatenated feature set demonstrates better discrimination capability than the individual feature vectors obtained from one source separately [22].

Furthermore, a number of decision-level fusion methods have been produced to find an alternative approach leading to a potential improvement in the classification performance. There are two categories of decision-level fusion techniques: methods operating on classifiers and methods operating on outputs.

Approaches in the methods operating on classifiers category generally put an emphasis on the development of the classifier structure [23]. They do not consider classifier outputs until the combination process finds single best classifier or a selected group of classifiers and only then their outputs are taken as a final decision or for further processing. On the other hand, approaches in the methods operating on outputs category operate mainly on classifiers outputs, in which the combination of classifiers outputs is calculated [24].

Both feature-level fusion and decision-level fusion were applied in this study in order to highlight the impact of using either of the two methods on the overall system performance. This was done in order to

find the best fusing method for producing a free-text keystroke authentication system with a low training requirement.

5 Experiments, Results and Discussion

This section presents the experiment results and discussion, in which the data collection, data space and the experimental results are indicated. A discussion about the experiment results and some comparisons with previous studies is performed in this section as well.

5.1 Data Space

A total of twenty-five users participated in this study's experimentation. They had different levels of typing skills that varied between moderate and very good. During data collection, the participants were asked to perform eight typing tasks each of which consisted of around 900 characters. The text included was an excerpt from the Guardian newspaper and it included both upper and lower case letters in addition to numbers and punctuation marks. Furthermore, the data was acquired in different sessions as the users were requested to complete each of the eight tasks in a separate session. 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.

The data collection was performed on a GUI program implemented using the C++ language. The application was downloaded on the users' personal machines to maximize their comfort as 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.

Although there were 75 timing features captured from each user's typing stream, 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. Indeed this was the reason for excluding four of the key-pair categories from the study. The omitted key-pairs, with less than 10 instances, are: Adjacent/DifferentSide, FourthAdjacent/RightSide, NonAdjacent/RightSide and NonAdjacent/LeftSide. Therefore 20 timing features corresponding to these key-pairs were excluded from the final feature set. Thus, a total of 55 timing feature were considered in the experimentation.

When observing the timing data collected in this experiment, a number of outliers occurred. Outlier data has been identified to be as much as three standard deviations above or below the mean as suggested by

Joyce and Gupta [7]. These particularly very large or very small data points were discarded from the final data as they were deemed to represent noise that might affect the overall system performance.

In addition, timing data was normalized before handing it to the machine learning technology [25]. Therefore, all the data was normalized to be between [0,1] to add a sense of uniformity to the data as attributes in greater numeric ranges might otherwise have dominated those in smaller numeric ranges [26].

The final step of data pre-processing for the timing features involved creating the timing vector and storing it in the database as the user's profile. This process was carried-out by dividing the data into 8 equal sections each of which represented a different typing sample.

This was done by extracting the feature vector (which included all the instances of that feature) for each of the 55 features from each typing task separately (there were eight typing tasks as mentioned before). Then, each feature vector from all the tasks are concatenated to produce 55 large vectors, one for each feature. These large vectors were stored in a feature matrix for each user.

Each of the large vectors was divided equally into 8 parts. The size of each part, among different features, varied depending on the number of times the key-pair associated with that timing feature appeared in the text. Although the number of each key-pair appearance may vary, it is fairly similar between participants as they all typed similar text. The mean of each feature among these 8 divisions is computed and then stored in the corresponding user's timing vector (V). Thus, there are eight timing vectors (Vs) for each user which were employed as the user's typing samples.

For non-conventional typing features, there was no need for outlier discarding as the features did not rely on a time factor that might add noise in the form of too large or too small time lags. Moreover, no scaling was needed as the non-conventional features are all quantities that represent percentages which vary between 0 and 1. Lastly the non-conventional typing features were calculated and stored in the non-conventional features vector at the database as the user's profile. Similar to the timing features, each one of the eight typing tasks was considered as a single typing sample.

5.2 Experiment and Results

As mentioned earlier, two types of fusion techniques are applied in this study. They are: feature-level fusion and decision-level fusion (as illustrated in Figure 4). A description of how each type of fusion was applied is presented in this section.

5.2.1 Feature-level Fusion

Timing features and non-conventional features were combined to produce a larger set of features. This allowed for the 55 timing features and the 9 non-conventional typing features to be joined in order to create a 64 mixed features dataset. This set was exploited in two classification processes. One of which was done using the SVMs/ACO technique and the other was done using decision trees (DTs).

SVMs and DTs were used in this experiment for comparison purposes as they are both simple and successful classification techniques [26,27]. These two classifiers were chosen as rivals because they follow completely different mechanisms for distinguishing classes [29]. In SVMs, the classification is performed by finding the optimum separating hyperplane [30], whereas classification in decision trees is purely based on decision rules [31].

Nonetheless, many pattern recognition and machine learning methods were utilized for classification in the area of keystroke dynamics. For example: Bayes classifier [32], K-nearest neighbour [33], Fuzzy logic [34] and Random Forests [35] were used. Moreover, Neural networks were also implemented for the purpose of distinguishing between the typing behaviour of different users, as in [11, 35]. Although using Neural networks yielded good results, it is considerably slow to apply and train.

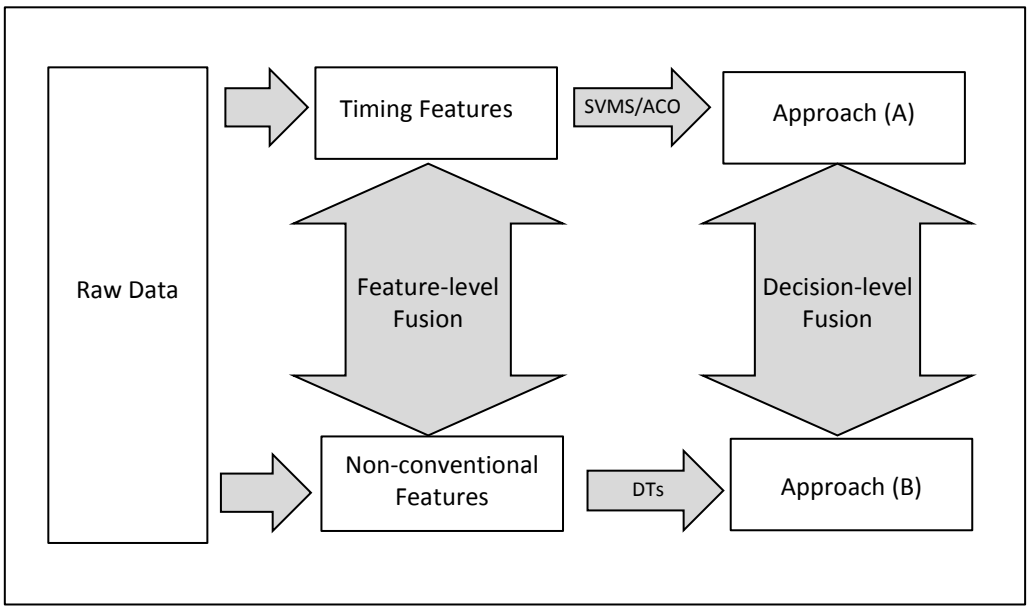

Fig. 4: An overview of the proposed algorithm.

Both methods, i.e. SVMs and DTs, were 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 [37]. N-fold cross-validation divides the data up into n chunks and trains it 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.

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 [10]. Equation 4 and 5 are used to compute FAR and FRR, respectively [38]. The final error rates resulting from the feature-level fusion are shown in Table 3.

$$\text{FAR} = \frac{\text{Number of false accept attempts (accepted imposters)}}{\text{Total number of impostor attempts}} = \frac{False\ Positive}{True\ Negative+False\ Negative} \quad (4)$$

$$\text{FFR} = \frac{\text{Number of false reject attempts (rejected legitimate users)}}{\text{Total number of legitimate attempts}} = \frac{False\ Negative}{True\ Positive+False\ Positive} \quad (5)$$

Table 3: Feature-level fusion of timing features and non-conventional features

	FAR	***FRR***
SVMs/ACO	0.0156	0.375
DTs	0.00896	0.215

5.2.2 Decision-level Fusion

The outputs of two approaches were fuse to achieve decision-level fusion in this study. The two approaches are described in this section.

Approach (A) utilized Support Vector Machines (SVMs) to classify the timing features. Having a large feature set, such as the timing features in this study, increases computational cost; in addition to raising the complexity of the classification process. Therefore, it was necessary to incorporate a feature subset selection mechanism to reduce redundancy [39]. Feature subset selection is considered as an optimization problem, in which the space of all possible features is scrutinized to find the feature or set of features that

produce optimal or near-optimal performance, i.e. those that minimize the classification error [40]. Ant Colony Optimization (ACO) proved to be a good candidate for achieving that goal [39]. Thus, ACO was applied to the timing features to select a features subset fed to the SVMs classifier. The RBF-kernel SVMs multiclass classification process was implemented on MATLAB with the aid of the LIBSVM library [41]. The LIBSVM library uses the one-against-one multiclass strategy. In this method, there are N(N +1)/2 classifiers built for N classes, one for each pair of classes [30]. After that voting is implemented to select the label with the most votes among the classifiers.

Radial Basis Function (RBF) kernel was used in this study as it succeeds in separating more complicated datasets compared with other kernels [42]. In addition, many studies suggests the use of the RBF as it nonlinearly maps samples into a higher dimensional space when there is a complicated relationship between the class labels and its attributes which fits to the keystroke dynamics problem [25].

SVMs parameters C and gamma were chosen using grid search and cross-validation. A range from 10-3 to 103 was set for both C and gamma values as recommended by the research done in [25]. All combinations of C and gamma were tested using a 10-fold cross-validation. The best performing value pair for C and gamma was chosen. The C & gamma pair chosen for this study was: C= 101, gamma=100.

In Approach (B), the non-conventional features were analysed using a DTs classifier. Building a decision tree depends on choosing which attribute to test at each node in the tree. A process that incorporates implicit feature subset selection, in which redundant features are excluded from the tree building process [43]. The tree structure, i.e. the order in which attributes were chosen to be tested at each node, differed each time a different training set was used. The Statistics toolbox in MATLAB was used to fit the tree and predict the class of each of the test data.

Both approaches were carried-out using cross-validation in the same manner described in the previous section. Additionally, as both Classifiers, from approaches (A) and (B), produced crisp, single class labels, we used the majority voting method to fuse the two classifiers. Majority voting was utilized because of its simplicity and effectiveness [44].

General voting follows the logic reported by Ruta and Gabrys [18]: let the output of the classifiers form the decision vector d defined as: $d = [d_1, d_2, \dots, d_n]^T$ where $d_i \in [c_1, c_2, \dots, c_m, r]^T$, c_i denotes the label of the i-th class and r is the rejection of assigning the input sample to any class. Let binary characteristic function be defined as follows:

$$B_j(c_i) = \begin{cases} 1 \text{ if } d_j = c_i \\ 0 \text{ if } d_j \neq c_i \end{cases} \quad (1)$$

Then the general voting routine can be defined as:

$$E(d) = \begin{cases} c_i \text{ if } \forall_{t \in \{1,\dots,m\}} \sum_{j=1}^{n} B_j(c_t) \leq \sum_{j=1}^{n} B_j(c_i) \geq \alpha . m + k(d) \\ r \text{ otherwise} \end{cases} \quad (2)$$

Where k(d) is a function that provides additional voting constraints and α= 0.5 is used in the majority vote [18].

Using cross validation, each of the 8 typing tasks was considered a single sample, each of which was used as a test sample in the eight cross validation experiments. This was done for both approaches yielding in 16 different cross-validation experiments. Each experiment's result was considered as a vote, i.e. there were 16 results to be included in the overall voting. Eight of these results belong to approach (A) and the other eight belong to approach (B). Using the majority voting scheme, the overall error rates resulting from this decision-level fusion yielded both 0.00 FAR and 0.00 FRR.

5.3 Discussion

This experiment attempts to improve two methods that were applied using timing features and non-conventional features to reduce the training requirement for free-text keystroke dynamics. In the first study (applying approach (A)), we exploited the timing features extracted from key-pairs and used ACO to select a feature subset that was fed into an SVMs classifier. The resulting FAR was satisfactory whilst the FRR was not. Moreover, in the second study (applying approach (B)), the non-conventional typing features were utilized to distinguish between users using the DTs classifier. Good FAR and FRR were produced yet the FRR was still higher than desired. Thus a fusion of these two methods was the next step to improve the user verification rate in order to achieve better authentication performance. Table 4 shows the error rates produced in the previous studies.

Table 4: Previous studies performance.

Study	*Approach*	*Features*	*Method*	*FAR*	*FRR*
Study1	Approach (A)	Timing feat.	SVMs/ACO	0.00245	0.384
Study 2	Approach (B)	Non-conv. feat.	DTs	0.0104	0.25

In feature-level fusion, SVMs/ACO produced average rates, yet it very slightly improved Approach (A)'s FRR. Meanwhile, DTs produced overall good error rates and it was able to slightly enhancing both error rates of Approach (B). Therefore, the fusion of the two feature sets had improved the FRR figure yet the FAR was only improved in the DTs case.

Furthermore, DTs proved to have better recognition outcome compared with SVMs/ACO; as shown in Table 3, DTs are able to produce lower error rates compared with SVMs/ACO. That is due to its ability to build a classification tree that finds the attributes returning the most homogeneous branches, i.e. choose the features that provide the best representation of human typing patterns [45]. This has a lot to do with the nature of the data in this case and its ability to convert into rules used by DTs to separate the data [46].

To improve the results further an ensemble of decision trees can be used to create a random forest classifier. Whereas a decision tree is built using the whole dataset considering all features, in random forests, each tree uses only a fraction of the features which are selected at random [47]. Therefore, a number of decision trees are grown with different subset and hence each decision tree will be different. Each tree will vote for a particular class and the class with the majority of votes is the predicted class. It is argued that the random forest is always at least as good in terms of model fit and stability to decision trees. But it's less intuitive and harder to interpret and fix [48].

It is noteworthy that both methods, i.e. SVMs and DTs, include some form of feature subset selection. ACO is a feature subset selection technique applied to the fused features to select the most suitable ones before being fed to SVMs [39]. DTs are also performing feature subset selection on the fused typing features when building and pruning the decision tree [49]. However, using all features with no feature selection did not produce good results as the level of noise was larger in such high features dimensionality [50]. The Curse of Dimensionality can be used to explain the performance deterioration when using a larger number of features. The Curse of Dimensionality corresponds to the problem that the amount of training needed grows exponentially with the number of features [51]. As we only have 8 samples per person, the number of features should be as small as possible to correctly represent the small number of samples.

When looking more closely at the features selected to be utilized in both methods, we found that non-conventional features were in the majority over timing ones. Four features out of five selected by ACO, in the first method, and six features out of nine chosen by DTs to be included in the tree building, in the second method, were non-conventional features. This supports the belief that non-conventional features

represent human typing patterns more precisely compared with timing features in a reduced training free-text keystroke system. The selected features for both methods are listed in Table 5. Consequently, non-conventional features appear to have a strong relationship between input values and target values, in this data set. A strong input-target relationship is formed when knowledge of the value of an input improves the ability to predict the value of the target which helps in understanding the characteristics of the target [29].

Table 5: Selected subsets of features.

	Timing feat.	***Non-conventional feat.***
SVMs/ACO	SL-H2	Error rate | RSA | LSA | LSB
DTs	AL-H1 | SL-UD | FL-H1	WPM | negUD | negUU | Error rate | RSA | LSA

In decision-level fusion, using majority voting has considerably improved the results of Approach (A) and Approach (B), individually. Perfect recognition rate was achieved using decision-level fusion. Even though not all votes were for the correct class, there was no common wrong class. The majority of votes were either for the correct class or for random classes which has no effect on the majority vote. Approach (B), i.e. DTs | non-conventional features, have contributed the most to this astounding results. It produced more correct votes which lead to correct overall decision. This supports our existing finding which state that non-conventional features represent the human typing behavior better than timing ones.

Combining classifiers have been shown to reduce the error rate in classification tasks opposite to single classifiers [3]. Moreover, combining different techniques to come up with a final decision makes the performance of the system more robust against the difficulties that each individual classifier may have on each particular data set [52]. This is due to different reasons such as statistical, computational or representational reasons [3]. In the case of free-text keystroke dynamics, applying fusion will help to improve the system performance while satisfying the low amount of training data requirement. This happens because the different classifiers handle the same data differently to obtain the predicted class. Which allows for the combining of power of the two classifiers to improve the final decision.

Using a fusion between SVMs and DTs which are both simple classifiers that perform training in a fast manner produces a fairly simple system [26, 27]. In contrast, other classifiers such as Neural networks produced a much more complex system as it is considerably slow to apply and train [12].

Figure 5: Comparison between the error rates produced by feature-level fusion and decision-level fusion.

A comparison between the error rates produced by feature-level fusion and decision-level fusion is illustrated in Figure 5. It demonstrates the low error rates produced by feature-level fusion in the case of the ACO/SVMs and DTs classifiers. It also shows a slight advantage of the DTs method over the ACO/SVMs. Moreover, the decision-level fusion produced zero FAR and FRR which proved that using majority voting decision-level fusion produces better system performance compared with the feature level-fusion and succeeds in producing a very good recognition system.

Comparing the performance of keystroke dynamics systems and therefore determining the method to follow for achieving the best authentication accuracy is not a straightforward task. Due to the variation of conditions that might be affecting the study participants, environment or procedure, the comparison between two or more methods is not always accurate [5]. This makes comparing the fusion methods followed in this study with other fusion performed in the literature a hard task. Nonetheless, two similar studies that involved a decision-level and a feature-level fusion, respectively, are presented here to give an idea of similar current research, as shown in Table 6. In addition, for comparison, some single-method keystroke dynamics studies are also presented in Table 6.

Table 6: Comparison with state-of-the-art studies.

Study	Participant no.	Characters no.	Fusion	Features		System performance		
				Conv.	*Non-conv.*	*Accuracy*	*FAR*	*FRR*
Davoudi & Kabir [53]	21	11700- 13500		√			0.001	0.188

Zahid et al. [54]	25	12500		√			0.292	0.308
Hempstalk et al. [55]	10	10800- 30000			√		0.113	0.331
Curtin et al. [15]	8	15000	√	√	√	0.985		
Ahmed & Traore [56]	53	11000	√	√			0.001	0.048
This study (Feature-level fusion)	25	7200	√	√	√	0.80	0.009	0.215
This study (Decision-level fusion)	25	7200	√	√	√	1.0	0.000	0.000

Decision-level fusing of the two methods suggested in the study conducted by Ahmed and Traore [56] has produced low error rates, yet the decision-level fusion proposed here succeeded in achieving zero error rates. The number of participants, however, is larger in the Ahmed and Traore study which might be a contributing factor. In addition, the accuracy produced from the feature-level fusing in this study is fairly similar to that produced from fusing the timing and non-conventional features, in the work conducted by Curtin et al. [15], despite the very low number of participants included in the latter study.

The main goal of this research which is relieving the users from the tedious training input to achieve a user-friendly authentication system. This was not followed in most of the studies reported literature. As illustrated in table 6, in the studies mentioned for comparison, long input was collected from users as the system required substantial amounts of data for training.

Although the work done by Davoudi & Kabir [53]and Ahmed & Traore [56] resulted in an FAR which was very close to zero, the fact that it was not exactly zero means that there was a percentage of individuals who were mistakenly identified as legitimate users. In our study, on the other hand, there were no unauthorized persons mistakenly identified as legitimate (thus a zero FAR was achieved). This showcases the superiority of the method used in this study compared with other methods found in the state of the art literature.

Moreover, the method introduced in this study used the least number of characters compared with previous studies, i.e. 7200 characters. This corresponds directly with the aim of the method introduced in this research, which is reducing the text used in order to relieve the users from tedious training input and thus to achieve a user-friendly authentication system.

The number of participants included in this study, i.e. 25 individuals, is considered very close to most of previous studies i.e. Davoudi & Kabir [53]: 21, Zahid et al. [54]: 25, etc. Which leads us to believe that the results produced by our studies' have the same level of credibility. That said, Ahmed & Traore's [56]

study was applied to a larger number of participants, i.e. 53, which increases the credibility of its results. It is recognized however that increasing the number of participants included in our research is necessary to provide further judgment about the trustworthiness of our method. This will be targeted in future work.

6 Conclusion

In this paper, fusion was applied to improve the performance of keystroke dynamics authentication whilst using the least amount of data for training. Moreover, a comparison between decision-level and feature-level fusion has been presented to improve free-text keystroke dynamics authentication. Feature-level fusion was performed to combine timing features and non-conventional typing features while decision-level fusion was carried-out to merge the outcomes of two classification approaches using the majority voting approach. In the first approach, SVMs were used to classify a subset of the timing features that was selected using ACO while the second method used DTs to classify non-conventional typing features.

Decision-level fusion proved to be superior to feature-level fusion as it succeeded in producing zero FAR and FRR. This dramatic enhancement shows the fact that using majority voting on the joint decision of the two approaches provides a good indication of the users' typing behaviour. This will eventually provide a good assessment of the user's identity. It also has been noted that non-conventional typing features have the edge over the timing features as they provided a better recognition rate throughout the study. The results produced by this study offer sufficient evidence that the fusion strategies proposed here is worthy of further study.

There is much more that can be done to improve on this study. One example of which is to investigate other feature subset selection techniques and classification methods. Experimenting with more fusion techniques might also contribute positively to the overall system performance.

Acknowledgments

The authors wish to extend their gratitude to the participants who were involved in this experiment for the time they took out of their busy schedules to contribute in this study.

References

[1] F. Monrose, A. Rubin, Authentication via keystroke dynamics, in: 4th ACM Conference on Computer and Communications Security, New York, 1997: pp. 48–56.

[2] R. Biddle, M. Mannan, P.C. Van Oorschot, T. Whalen, User Study , Analysis , and Usable Security of Passwords Based on Digital Objects, IEEE Transactions on Information Forensics and Security. 6 (2011) 970–979.

[3] F. Moreno-Seco, J.M. Inesta, P.J.P. de Leon, L. Mico, Comparison of classifier fusion methods for classification in pattern recognition tasks, in: Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Springer, Berlin, 2006.

[4] A. Alsultan, K. Warwick, Keystroke Dynamics Authentication : A Survey of Free-text Methods, International Journal of Computer Science Issues. 10 (2013) 1–10.

[5] S.P. Banerjee, D.L. Woodard, Biometric authentication and identification using keystroke dynamics : A survey, Journal of Pattern Recognition Research. 7 (2012) 116–139.

[6] J. Kim, H. Kim, P. Kang, Keystroke dynamics-based user authentication using freely typed text based on user-adaptive feature extraction and novelty detection, Applied Soft Computing. (2017).

[7] R. Joyce, G. Gupta, Identity authentication based on keystroke latencies, Communications of the ACM. 33 (1990) 168–176. doi:10.1145/75577.75582.

[8] E. Lau, X. Liu, C. Xiao, X. Yu, Enhanced User Authentication Through Keystroke Biometrics, in: Computer and Network Security, Massachusetts Institute of Technology, 2004.

[9] H.R. Lv, Z.L. Lin, W.J. Yin, J. Dong, Emotion recognition based on pressure sensor keyboards, in: IEEE International Conference on Multimedia and Expo, Ieee, 2008: pp. 1089–1092. doi:10.1109/ICME.2008.4607628.

[10] D. Gunetti, C. Picardi, Keystroke analysis of free text, ACM Transactions on Information and System Security. 8 (2005) 312–347. doi:10.1145/1085126.1085129.

[11] J. Hu, D. Gingrich, A. Sentosa, A k-nearest neighbor approach for user authentication through biometric keystroke dynamics, in: IEEE International Conference on Communications, Beijin, 2008: pp. 1556–1560.

[12] D. Raghu, C.R. Jacob, Y.V.K.D. Bhavani, Neural network based authentication and verification for web based key stroke dynamics, International Journal of Computer Science and Information Technologies. 2 (2011) 2765–2772.

[13] P.S. Teh, A.B.J. Teoh, T.S. Ong, H.F. Neo, Statistical Fusion Approach on Keystroke Dynamics, 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System. (2007) 918–923. doi:10.1109/SITIS.2007.46.

[14] S. Hocquet, J. Ramel, H. Cardot, Fusion of methods for keystroke dynamic authentication, in: Fourth IEEE Workshop on Automatic Identification Advanced Technologies, 2005: pp. 224–229.

[15] M. Curtin, C. Tappert, M. Villani, G. Ngo, J. Simone, H.S. Fort, S. Cha, Keystroke biometric recognition on long-text input: a feasibility study, in: IMECS, 2006.

[16] D. Gunetti, C. Picardi, G. Ruffo, Keystroke analysis of different languages : A case study, in: Advances in Intelligent Data Analysis VI, 2005: pp. 133–144.

[17] H. Davoudi, E. Kabir, A new distance measure for free text keystroke authentication, in: 14th International CSI Computer Conference, Ieee, 2009: pp. 570–575. doi:10.1109/CSICC.2009.5349640.

[18] D. Ruta, B. Gabrys, An overview of classifier fusion methods, Computing and Information Systems. 7 (2000) 1–10.

[19] J. Bezdek, J. Keller, R. Krisnapuram, N. Pal, Fuzzy models and algorithms for pattern recognition and image processing, Springer, 1999.

[20] J. Esteban, A. Starr, R. Willetts, P. Hannah, P. Bryanston-cross, A review of data fusion models and architectures: towards engineering guidelines, Neural Computing & Applications. 14 (2005) 273–281.

[21] A. Ross, R. Govindarajan, Feature level fusion using hand and face biometrics, in: SPIE Conference on Biometric Technology for Human Identification, Orlando, 2005: pp. 196–204.

[22] H. Lv, W. Wang, Biologic verification based on pressure sensor keyboards and classifier fusion techniques, IEEE Transactions on Consumer Electronics. 52 (2006) 1057–1063.

[23] T.H. Ho, J.J. Hull, S.N. Srihari, Decision Combination in Multiple Classifier System, IEEE Transactions on Pattern Analysis And Machine Intelligence. 16 (1994) 66–75.

[24] P.D. Gader, M.A. Mohamed, J.M. Keller, Fusion of handwritten word classifiers, Pattern Recognition Letters. 17 (1996) 577–584.

[25] C.W. Hsu, C.C. Chang, C.J. Lin, A practical guide to support vector classification, Tech. Rep., Department of Computer Science, National Taiwan University. (2003).

[26] R.A. Irizarry, B. Hobbs, Y.D. Beazer-barclay, K.J. Antonellis, U.W.E. Scherf, T.P. Speed, Exploration , normalization , and summaries of high density oligonucleotide array probe level data, Biostatistics. 4 (2003) 249–264.

[27] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery. 2 (1998) 121–167.

[28] M. Friedl, C. Brodley, Decision tree classification of land cover from remotely sensed data, Remote Sensing of Environment. 61 (1997) 399–409.

[29] K. Warwick and R. Craddock, An introduction to radial basis functions for system identification. A comparison with other NN, (n.d.).

[30] T. Fletcher, Support vector machines explained, UCL. (2009) 1–19. doi:10.1002/9780470503065.app2.

[31] S.R. Safavian, D. Landgrebe, A survey of decsion tree clasifier methodology, IEEE Transactions on Systems, Man, And Cybernetics. 21 (1990) 660–674.

[32] M.S. Obaidat, B. Sadoun, Verification of computer users using keystroke dynamics, IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics : A Publication of the IEEE Systems, Man, and Cybernetics Society. 27 (1997) 261–9. doi:10.1109/3477.558812.

[33] N. Bakelman, J. V. Monaco, S.H. Cha, C.C. Tappert, Continual keystroke biometric authentication on short bursts of keyboard input, in: Pace University CSIS Research Day, 2012: pp. 1–7.

[34] W.G. d. Ru, J.H.P. Eloff, Enhanced password authentication through fuzzy logic, IEEE Expert. 12 (1997) 38–45.

[35] N. Bartlow, B. Cukic, Evaluating the reliability of credential hardening through keystroke dynamics, in: The 17th International Symposium on Software Reliability Engineering, Ieee, 2006: pp. 117–126. doi:10.1109/ISSRE.2006.25.

[36] A.A.E. Ahmed, I. Traore, A. Almulhem, Digital fingerprinting based on keystroke dynamics, in: The Second International Symposium on Human Aspects of Information Security and Assurance, 2008.

[37] P. Refaeilzadeh, L. Tang, H. Liu, Cross-Validation, Encyclopedia of Database Systems. (2009) 532–538.

[38] Q. Zou, L. Ni, Q. Wang, Q. Li, S. Wang, Robust Gait Recognition by Integrating Inertial and RGBD Sensors, IEEE Transactions on Cybernetics. (2017).

[39] M. Karnan, M. Akila, Personal authentication based on keystroke dynamics using soft computing techniques, in: Second International Conference on Communication Software and Networks, Ieee, Singapore, 2010: pp. 334–338. doi:10.1109/ICCSN.2010.50.

[40] M. Karnan, M. Akila, N. Krishnaraj, Biometric personal authentication using keystroke dynamics: A review, Applied Soft Computing. 11 (2011) 1565–1573. doi:10.1016/j.asoc.2010.08.003.

[41] C. Chih-Chung, C.-J. Lin, Libsvm, ACM Transactions on Intelligent Systems and Technology. 2 (2011) 1–27. doi:10.1145/1961189.1961199.

[42] A. Vlachos, Active learning with support vector machines, Masters Dissertation, University of Edinburgh. (2004).

[43] O. Maimon, L. Rokach, Data mining and knowledge discovery handbook, Springer, 2010.

[44] U. Bhattacharya, B.B. Chaudhuri, A majority voting scheme for multiresolution recognition of handprinted

numerals, in: The Seventh International Conference on Document Analysis and Recognition (ICDAR03), 2003: pp. 16 – 20.

[45] B.D. Ville, P. Neville, Decision trees for analytics using SAS enterprise miner, SAS Institute, 2013.

[46] E. Chen, Choosing a machine learning classifier, (2011). http://blog.echen.me/2011/04/27/choosing-a-machine-learning-classifier/ (accessed January 14, 2016).

[47] L. Zhang, P.N. Suganthan., Random forests with ensemble of feature spaces, Pattern Recognition. 47 (20014) 3429–3437.

[48] L. Zhang, R. Ye, P.N. Suganthan., Towards generating random forests via extremely randomized trees, International Joint Conference on Neural Networks (IJCNN). (2014).

[49] C. Ratanamahatana, D. Gunopulos, Scaling up the naive bayesian classifier : Using decision trees for feature selection, in: Workshop Data Cleaning and Preprocessing (DCAP '02), at IEEE International Conference on Data Mining (ICDM '02), 2002.

[50] Y. Saeys, I. Inza, P. Larrañaga, A review of feature selection techniques in bioinformatics, Bioinformatics. 23 (2007) 2507–2517. doi:10.1093/bioinformatics/btm344.

[51] A.K. Jain, R.P.W. Duin, J. Mao, Statistical Pattern Recognition : A Review, IEEE Transactions on Pattern Analysis And Machine Intelligence. 22 (2000) 4–37.

[52] W. Li, H. Jian, Y. Lizhi, A classifier fusion method based on classifier accuracy, in: IEEE International Conference on Mechatronics and Control (ICMC), 2014.

[53] H. Davoudi, E. Kabir, User authentication based on free text keystroke patterns, in: The 3rd Joint Congress on Fuzzy and Intelligent Systems, 2010.

[54] S. Zahid, M. Shahzad, S.A. Khayam, M. Farooq, Keystroke-based user identification on smart phones, in: The 12th International Symposium on Recent Advances in Intrusion Detection (RAID '09), 2009: pp. 224–243.

[55] K. Hempstalk, E. Frank, I.H. Witten, One-class classification by combining density and class probability estimation, in: The European Conference on Machine and Learning and Principles and Practice of Knowledge Discovery in Database, 2005: pp. 505–519.

[56] A. a. Ahmed, I. Traore, Biometric recognition based on free-text keystroke dynamics, IEEE Transactions on Cybernetics. 44 (2014) 458–472.