Bolstering user authentication: a kernel-based fuzzy-clustering model for typing dynamics

Original Scientific Paper

Anthony Metumaraibe Nwohiri
University of Lagos,
Faculty of Science, Department of Computer Sciences
University Road, Akoka-Yaba, Lagos Nigeria
anwohiri@unilag.edu.ng

Ufuoma Cyril Ogude
University of Lagos,
Faculty of Science, Department of Computer Sciences
University Road, Akoka-Yaba, Lagos Nigeria
uogude@unilag.edu.ng

Hai Vinh Nguyen
Vietnam National University,
Faculty of Mathematics, Mechanics and Informatics, Department of Informatics
Nguyen Trai Rd, Hanoi, Vietnam
nguyenhaivinh@vnu.edu.vn

Edilberto Moreno Sanchez
Autonomous National University of Mexico,
Astronomy Institute, Department of Information Technology and Scientific Computing
Coyoacán borough, Mexico City, Mexico
edilberto@astro.unam.mx

Abstract *– In most information systems today, static user authentication is accomplished when the user provides a credential (for example, user ID and the matching password). However, passwords appear to be the most insecure authentication method as they are vulnerable to attacks chiefly caused by poor password hygiene. We contend that an additional, non-intrusive level of security can be achieved by analyzing keystroke biometrics and coming up with a unique biometric template of a user's typing pattern. The paper proposes a new model for representing raw keystroke data collected when analyzing typing biometrics. The model is based on fuzzy sets and kernel functions. The corresponding algorithm is developed. In the static authentication problem, our model demonstrated relatively higher performance than some classic anomaly-detection algorithms, such as Mahalanobis, Manhattan, nearest neighbor, outlier counting, neural network, and the support-vector machine.*

Keywords*: anomaly detection, data mining, fuzzy clustering, keystroke biometrics, kernel function, machine learning, static authentication*

1. INTRODUCTION

Internet proliferation has exploded over the past decade. This has made efficient access control (AC) for information systems ever more challenging. Intruders may gain access to systems from virtually anywhere over the Internet. As a fundamental concept in security, AC regulates who or what can view or use resources in a computing environment. It restricts access to computers, networks, applications, files and other sensitive data. One of the crucial functions of AC systems is to ensure that "someone" is who they claim to be (authentication) and that they have the appropriate data access (authorization). Hence, user authentication is a major important challenge.

The user authentication problem can be tackled in many ways. In modern information systems, passwords remain the most common digital authentication method [1–4]. Passwords are typically a string of characters used to confirm a user's identity during the authentication process. While passwords are a weak form of protection, their simplicity makes them easy to use and administer. However, passwords are among the most vulnerable authentication methods. Poor password hygiene is a top cause of data breaches. A dictionary attack can be used to pick up the passphrase. Password hash can be leaked directly from their storage location to be cracked offline [4]. Digital signature systems (DSSs), also used for authentication, are free from dictionary attacks [5]. However, DSSs cannot guarantee secure storage of private keys because the keys are stored using other access control means [6, 7]. Moreover, digital signature needs to be verified and there is no legal backup for this verification process [6].

In view of the above, most high-security systems use biometrics to identify and authenticate individuals. The primary premise of biometric authentication is that any user can be precisely identified by intrinsic physical or behavioral traits. This authentication method comes with several benefits – it is convenient, every user has access to a unique set of biometrics, biometrics are hard to steal, and of course this technique comes with high security and assurance.

Biometric authentication uses unique biological traits to verify that someone is who they say they are. Such traits include palm print, fingerprints, hand geometry, voice, retinal patterns, iris recognition, facial recognition, DNA, odor/scent, and hand patterns [8]. Biometric systems can be divided into two categories: (a) physical biometrics systems – are effective, but quite expensive as they require special hardware; (b) behavioral biometrics systems – they analyze parameters such as a user's keystrokes dynamics, navigational patterns, screen pressure, typing speed, mouse or mobile movements, gyroscope position and more [9-11]. They do not require any special hardware and are easy to implement.

In this paper we first present the problem at hand, and look at existing user authentication methods. After that, we propose a keystroke data representation model and develop the corresponding algorithm. The algorithm is compared with some classic anomaly detectors.

2. PROBLEM STATEMENT

This research deals with the static authentication problem. Static authentication reuses a specific authenticator (e.g., static password). This type of authentication only provides protection against attacks in which an imposter cannot obtain the authenticator. An authentication process is strong if it is difficult to guess or decrypt the authenticator values and if the values themselves are secured in transit and while stored on the system [12].

The method works on a known pattern or other predefined text. The data (e.g., password) entered by the user when attempting to login is collected and compared with previous successful login attempts. This technique is an extension of the standard user ID/password-based authentication method (i.e., the system checks not only what the user typed, but how it was typed).

Several static authentication features are worth noting. First, the input data is relatively small. Typically, static authentication works in tandem with password authentication. This practically eliminates the use of extremely long passwords (over 100 characters) that the user would have to manually type.

Another feature is that input data is static in nature. The password, for the most part, remains the same from the moment it is created till it is changed or updated. In this case, only a small amount of typing biometrics data can be extracted. Besides, more often than not, the password is changed very rarely, which leads to a large sample. Therefore, static authentication should be optimized to be able to recognize a user based on a small set of parameters.

Static authentication should be completed as quickly as possible since the user will not be granted access until authentication data has been successfully processed. Until authentication is completed, the user will not be allowed to access the system. Therefore, the time interval between the moment the user enters his password and the moment he gains access to the system should be minimized as much as possible.

So, to put it more formally, our authentication problem can be described as follows. We assume there is a set of users performing certain keyboard actions, for example, pressing a key or typing a password. Here, the training task is to match a certain function (model) to each user; the model is to serve as a measure of the anomalousness of user actions, i.e., it will be used to verify the identity of real users and detect anomalous users. The authentication task is to measure (based on the model) the anomalousness of a new user action and process the calculated value. This value determines whether user authentication would be accepted or rejected.

The false rejection rate (FRR) and false acceptance rate (FAR) were used to evaluate obtained results. False rejection occurs when an authentic user is rejected by the system, while false acceptance is when an imposter is accepted. A lower FRR means less rejection and easier access by genuine users. A lower FAR indicates less imposter accepted. [13-15].

Authentication algorithms, including the one proposed in this paper, do not return a binary value – authentication successful/authentication failed. Rather, they return some real value indicating how well the authentication attempt matched the training data. Hence, it is necessary to introduce a certain threshold that would distinguish between accepted and rejected authentication attempts. Varying this real value, we can find a threshold at which FRR and FAR are equal. That is, the authentication system is configured such that the rate of false negatives and the rate of false positives are approximately equal. The crossover error rate (CER) describes the point where FRR and FAR are equal. It is one of the most important indicators used in evaluating the performance of any biometric security system [16, 17]. The CER describes the overall accuracy of a biometric system. Figure 1 shows that the lower the CER value, the higher the accuracy of the biometric system. If the threshold of acceptance (sensitivity) of the system is increased, FRR will increase and FAR will fall. Likewise, choosing a low threshold will result in high FAR and low FRR [15, 18].

Since the data used for the experiment contains password typing biometrics from many people, it can be assumed that the algorithm's performance will vary depending on the subject. Obviously, a highly efficient algorithm should produce equally good results regardless of the subject being authenticated. The greater the CER spread for different subjects, the harder it would be using the algorithm in practice. Therefore, the standard deviation of CERs for different subjects becomes another important parameter for evaluating the algorithm's performance.

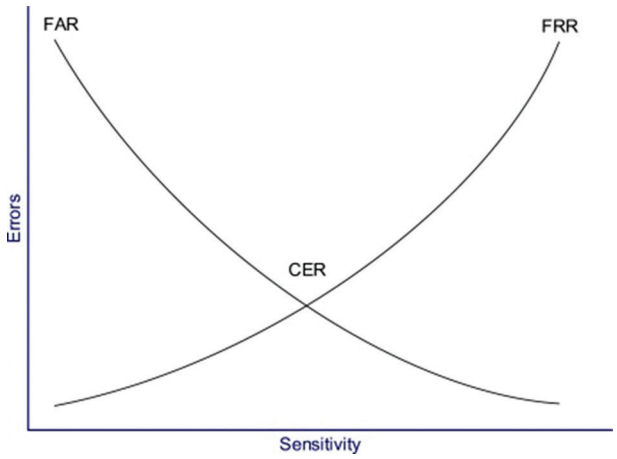

Fig. 1. False acceptance rate versus false rejection rate

3. EXISTING AUTHENTICATION APPROACHES

Existing authentication methods are based on various features collected while the user is typing at a computer keyboard. They are also based on the models (used to verify the real user's identity) created using these features.

3.1. DWELL TIME

Dwell time is the period during which a key is in a pressed state. As described by Wong et al. [19], the key hold method takes a vector as a model, the vector consists of some elements. Each of the elements correspond to a key on the keyboard, representing a pair – the mean dwell time of the key and the standard deviation for that key. So, the dwell time of a key (user action) is considered abnormal if the difference between it and the mean dwell time for that key is greater than the standard deviation for that key. A percentage of allowed abnormal actions is given. If this threshold is exceeded, authentication will be rejected. The authors achieved FRR <10% and FRR <10% all at a time.

3.2. KEY EVENT ORDER

Lau et al., [20] observed that when typing at the keyboard, some people sometimes unconsciously press a second key before releasing the first one. The authors called this phenomenon a "swap". A user model can be constructed by observing the key-press/key-release sequence and considering the number of "swaps". To obtain an anomaly score, the distance between the tested and the user set was compared with the average distance between the keystrokes of the same user. An authorization attempt was rejected whenever the distance went beyond one standard deviation. The resulting FAR and FRR depended strongly on pairs of users, with error rates ranging from 0% to 70%. This clearly makes the method unsuitable for use unless with other approaches.

3.3. RELATIVE TYPING SPEED

It is assumed that for each pair of keys, the typing speed remains about the same regardless of the text being typed. Therefore, it was suggested that the typing speeds of pairs of keys be measured and used as a user model. The distance between vectors of key pairs, ordered by typing speed, as proposed by Bergadano et al. [21], was used to build a model. Based on the resulting mean distance between any two vectors of the same user and between any two vectors of different users, the researchers suggested that an authentication attempt should be accepted only when the difference in vectors is less than 0.33 and rejected when it is greater than 0.66.

3.4. THE *SHIFT* KEYS

In their paper, Lau et al. [20] argued that people use the right and left *Shift* keys differently, and that this could be used for authentication. The authors divided users into 4 groups: strictly left-*Shift* users, strictly right-*Shift* users, those who used the left *Shift* key more often than the right, and those who used the right *Shift* key more often than the left. Obviously, if a user hits the expected group, that does not necessarily mean that any authentication attempt made by that user must be accepted since we have a limited number of groups (only 4), and the false acceptance rate is very high. However, when a user hits the wrong group, that authentication attempt will be rejected.

3.5. SHORT ALPHABETIC OR NUMERIC PASSWORDS

Techniques proposed in [22] and [23] used keystroke times as a model. However, in [22], times were measured using three different typologies, as shown in Table 1.

Table 1. Typologies of time.

Time topologies	Description
Absolute time	Consists of the dwell time (how long a key was held pressed) and the flight time (the duration between the moment the key was released and the moment the next one was pressed)
Cumulative time	Consists of the accumulated absolute times for typing a particular phrase. This allows to smooth out outliers.
Ratio time	This is the ratio of the dwell time to the flight time

A multiclass linear support vector machine (SVM) was used as the training algorithm, as it demonstrates high results on simple-structure data [24].

During data collection, the subjects were divided into two groups: those who were informed about the experiment and those who were not. It was demonstrated that obtained results were a function of users' awareness of the experiment. Specifically, the obtained FAR and FRR (1%-3%) were 3-5 times lower among the informed users.

4. PROPOSED APPROACH

4.1. DESCRIPTION OF THE MODEL AND ALGORITHM USED

Keystroke time intervals are the main inputs of the model. The various time intervals used for the feature space of our algorithm are presented in Table 2. The feature space was chosen as follows. Each password typing attempt is represented as a vector of time lapses between different key-press and key-release events. A key-press event occurs when a key that produces a character value is pressed down, while a key-release event occurs once the key is released.

Keystroke time intervals are the main inputs of the model. The various time intervals used for the feature space of our algorithm are presented in Table 2. The feature space was chosen as follows. Each password typing attempt is represented as a vector of time lapses between different *key-press* and *key-release* events. A *key-press* event occurs when a key that produces a character value is pressed down, while a *key-release* event occurs once the key is released.

Table 2. Timing vector for password-input events.

Time intervals	Description
Dwell time	The period, during which a key is in a pressed state. In other words, it is the length of time a key is pressed until it is released.
Press–press	Interval between two successive key presses (always positive)
Release–press	Interval between a key release and the next key press time (may be negative if next key is pressed before previous key is released)
Release–release	Interval between two successive key releases (always positive)

It should be noted that for the first key pressed, only the dwell time is measured.

We used the outlier detection technique presented in [25] as the static authentication problem. The method is based on two main ideas: using kernel functions to define distances and deploying the fuzzy set theory to build a user model. We have adapted the approach to our problem.

4.1.1 KERNEL FUNCTIONS

Kernel functions (KFs) provide a way to manipulate data. The function of kernel is to take data as input and transform it into the required form. Kernels represent a method of computing the dot product of two vectors in a certain feature space. They are widely used in various machine learning algorithms [26-30] and have been shown to be very efficient in tackling various attack/intrusion detection problems [31-33]. KFs allow for efficiency in biometric security systems. These functions enable you to avoid the trouble of having to go into an infinite-dimensional space; they also save you the time that would have been spent on computing map functions.

Nonlinear mapping from an input space of objects to the feature space is central in kernel methods. The kernel trick is a simple method which involves performing the mapping and the inner product simultaneously by defining its associated KF. The KF, see (1), computes, and returns the inner product between two inputs in the feature dimension.

$$K(x,y) = f(x) \cdot f(y) \tag{1}$$

Here *K* is the kernel function, *x* and *y* are *n*-dimensional inputs, *f* is a feature map from *n*-dimension to m-dimension space, *x·y* denotes a dot product. Usually, *m* is much larger than *n*. The Hilbert space serves as our *m*-dimension space.

Kernel function *K(x,y)* measures the distance (similarity) between two input objects *x* and *y*. This metric can be used to build distance functions. Kernels provide a way of computing dot products in some feature space without even knowing what this space is and what the map *f* is. So, there is no need to compute *f(x)* and *f(y)*; moreover, mapping is implicitly determined by *K(x,y)*. Consequently, computing time and memory costs is also not needed. This is where the basic advantages of kernel functions come in.

While classical kernel-based clustering algorithms are based on a single kernel, in practice it is often desirable to base clustering on combination of multiple kernels [34]. The use of different kernels adds a certain flexibility to our approach. It also expands possible configurations for the method, which can be selected such that optimal results are achieved.

In our approach, the following distance function based on the kernel function is considered:

$$d(x,y) = \sqrt{K(x,x) - 2K(x,y) + K(y,y)} \tag{2}$$

We will focus more on the use of dot products (see expression 1) as kernels and the use of the Gaussian kernel (see expression 3).

$$K(x,y) = \frac{e^{-(x-y)^2}}{2\sigma^2} \tag{3}$$

In expression (3), sigma σ is the standard deviation of the Gaussian distribution. It basically controls how "fat" the kernel function is going to be. It controls the variance around a mean value of the Gaussian distribution (how closely the values of a data set are clustered around the mean). As σ becomes larger, the more variance (allowed around mean) can be chosen to achieve the best results. Conversely, as σ becomes smaller, the less variance allowed around mean can be chosen to achieve the best results. The Gaussian kernel

transforms the dot product in the infinite dimensional space into a Gaussian function of the distance between points in the data space.

4.1.2 FUZZY CLUSTERING IN FEATURE SPACE

Introduced independently by Lotfi A. Zadeh and Dieter Klaua in 1965 [35, 36], fuzzy sets were an extension of the classical notion of set [37]. They are objects with a continuum of grades of membership. These sets are characterized by a membership function that maps from the universal set to a value between 0 and 1.

In our proposed method, we search for one common fuzzy cluster containing images of all objects from the original space *X*. **T**he degree of membership of the image of an object from *X* quantifies the grade of membership of that object to each fuzzy cluster, i.e., a value inverse to the anomaly. Images with "small" membership grades (less than a threshold established for a user) will be considered as illegitimate authentication attempts.

Petrovsky [25] demonstrated that a search for a fuzzy cluster in a feature Hilbert space, as we suggested above, results into the following fuzzy clustering problem

$$\left.\begin{array}{c}\min\limits_{U\in[0,1]^N,\ a\in H} J(\boldsymbol{D},\boldsymbol{c})\ ,\\ (D,c)=\sum_{i=1}^{N}(D_i)^m(f(x_i)-c)^2-\eta\sum_{i=1}^{N}(1-d_i)^m\end{array}\right\} \quad (4)$$

where *H* is the feature space containing vectors representing authorization attempts; c is the center of the fuzzy cluster in the feature space corresponding to legitimate user authorization attempts; *D* is the function of the membership degree vector; *N* is the number of legitimate authorization attempts used for training; $d_i \in [0,1]$ is the membership degree of image $f(x_i)$ with respect to the fuzzy cluster in the feature space, and, accordingly, the typicalness degree of object xi; m is the fuzziness degree, and eta (η) is the distance from the cluster center, where the typicalness degree of the object is considered to be 0.5.

In our method, unlike in some classic fuzzy clustering methods, the center of the cluster or the values $f(x_i)$ cannot be clearly expressed. Nonetheless, $J(D,c)$ can be minimized using the following iteration algorithm based on randomized block-coordinate descent [38–40].

First, *D* and η are initialized: two points in the training set at maximum distance from each other are found; η is taken equal to the square of the distance between these two points and does not change throughout the algorithm; elements of D are taken equal to each other, $d_i^0 = 1/N$; that is, the cluster center is the same with the "center of gravity" of the images of points *x*. For such a selection of η, the typicalness degree of the objects from the learning set is always greater than 0.5.

Second, the cluster center is then calculated.

$$c \cdot c = \left(\sum_{j=1}^{N} d_i^m \sum_{i=1}^{N} d_i^m K(x_i, x_j)\right) \Big/ \left(\sum_{i=1}^{N} d_i^m\right)^2 \quad (5)$$

Third, the distance to the new cluster center is computed for all $j \in [1,N]$.

$$f(x_j)\cdot c = \left(\sum_{i=1}^{N} d_i^m K(x_i, x_j)\right) \Big/ \left(\sum_{i=1}^{N} d_i^m\right) \quad (6)$$

Fourth, new degrees of membership of training vectors are computed for all $j \in [1,N]$:

$$d_j = \frac{1}{1+\left(\frac{c\cdot c + K(x_i,x_j) - 2(f(x_j)\cdot c)}{\eta}\right)^{m-1}} \quad (7)$$

The second, third and fourth steps are repeated until:

$$\|D^l - D^{l-1}\| < \varepsilon \quad (8)$$

where *l* is the step number, ε is the required accuracy.

In this case, anomaly function *F(x,X)*, which calculates the typicalness degree of a new object takes the form.

$$F(x,X) = \frac{1}{1+\left(\frac{c\cdot c + K(x,x) - 2(f(x,X)\cdot c)}{\eta}\right)^{m-1}} \quad (9)$$

where

$$\boldsymbol{f(x,X)\cdot c} = \left(\sum_{i=1}^{N} \boldsymbol{d_i^m K(x_i, x)}\right) \Big/ \left(\sum_{i=1}^{N} \boldsymbol{d_i^m}\right) \quad (10)$$
$$\boldsymbol{d_1 \ldots d_N \in X, |X| = N\P}$$

In this method, η was chosen as the square of the distance between two points at maximum distance from each other in the training set. It does not change throughout the algorithm. We call this a simplified variant.

However, there is a more complex way of choosing η. The square of the distance from the center of the cluster to the farthest non-outlier vector is used to estimate the cluster radius at each iteration. Outliers are suggested to be the fraction of vectors farthest from the cluster center, which is a parameter of the algorithm. The degree of membership of an object image to the fuzzy cluster in the feature space may be viewed as a typicalness degree of the object. In this case, typicalness degree *F(x,X)* will be > 0.5 if *x* lies inside the cluster, < 0.5 if it lies outside the cluster, or equal to 0.5 if it lies on the border of the cluster. Therefore, in implementing the model, 0.5 is used as the initial minimum typicalness degree by which a typing attempt is to be considered legitimate.

The simplified variant comes with some merits. Its basic advantage is that the anomaly function is continuous, which makes it possible to compare typicalness degrees of objects and also to modify the outlier factor criterion without reconstructing the model. Moreover, the proposed algorithm is considerably simpler than those used for solving quadratic programming problems [25]. The complexity of the algorithm itself is linear.

However, calculation of the kernel matrix has $O(n^2)$ time complexity, where n is the size of the learning set [41].

4.1.3 DATA PRE-PROCESSING

Features in the collected data may be heterogeneous, and their values may have different bounds. Therefore, given the peculiarities of the suggested algorithm, the data should be normalized, bringing the range of values to common boundaries for all the features. The best normalization method for this problem was chosen experimentally on a standard dataset – normalization to the absolute deviation value. Suppose that a feature *p* is encountered in training for *N* password-typing attempts. Then for *p*, the normalization factor for vector *x* would be:

$$W_p = \sum_{i=1}^{N} \frac{|x_i - \bar{x}|}{N}, \quad x'_p = \frac{x_p}{W_p} \tag{11}$$

where $\bar{x}$ is the arithmetic mean of the elements of *x*, and *x'* is the normalized vector of *p*.

Among other possible normalization factors that can be used are the square root of the above value, the interquartile range, and some other factors. However, as would be shown later, absolute deviation gives the best results.

5. EXPERIMENT

We conducted a series of experiments in order to compare the suggested method with existing ones and select optimal classification parameters. So, the proposed algorithm was implemented in *R* – a programming language and free software environment for statistical computing and graphics [42].

5.1. EXPERIMENTAL DATA AND SET-UP

In order to be able to compare the performance of our algorithm with those of other algorithms, we had to conduct an experiment using the same conditions as those used in other methods. The conditions involved having the same type of data, the same amount of training data and test data. Experimental data was obtained from a study by Killourhy et al. [43]. The reason for this is, the data is consistent with our formulated static authentication problem, it is representative enough.

We had to also take the experimental set up parameters from the same source. Enrolled for the study were 51 subjects (26 males, 25 females; 35 right-handed and 16 left-handed subjects). They completed eight data-collection sessions (of 50 passwords each), making it 400 password-typing samples in total. We collected password typing data from the 51 subjects who each typed 400 repetitions of a password. We then extracted various timing features, such as dwell time, *press–press* time, *release–press* time, *release–release* time, etc.

Moving further, some assumptions were made. We considered a situation where a user's long-time password has been compromised by an impostor. We assume that the legitimate user is practiced in typing his/her password, while the illegitimate user is not (for example, he is typing it for the first time). So, in this case, we measure how well the detection algorithm is able to differentiate between the genuine user's typing and the impostor's typing.

For a start, we designate one of the 51 subjects as the legitimate user, and the rest as illegitimate users. We train our detector and test its ability to identify the genuine user and impostors. So, the training phase of the algorithm is run on the timing vectors from the first 200 password repetitions typed by the genuine user. The algorithm builds a model of the user's typing behavior. Then, the test phase of the algorithm is run on the timing vectors from the remaining 200 password repetitions typed by the genuine user. The anomaly scores assigned to each timing vector are recorded as legitimate user scores. We then run the test phase of the detection algorithm from the first five password repetitions typed by each of the 50 illegitimate users. The anomaly scores assigned to each timing vector are recorded as illegitimate user scores. In total we have 450 attempts.

The above process is repeated, each time designating one of the other subjects as the legitimate user in turn. After training and testing our algorithm, we have a total of 51 sets of legitimate user and illegitimate user scores.

This experimental model corresponds rather precisely to the real scenario of using user authentication based on keystroke dynamics analysis: the first attempts are used for training, assuming that training occurs at the moment when the password changes to a new one, when a legitimate user is just beginning to develop his characteristic password typing traits. For detection, the last attempts are used, where the legitimate user exhibits the developed password typing traits, while the illegitimate ones, being previously unfamiliar with the password, do not.

5.2. PARAMETER SELECTION AND RESULTS

In order to evaluate the influence of all parameters and select the values that best fit the problem, several series of experiments were conducted where we varied parameter values within given intervals. Results were evaluated and appropriate conclusions on how a particular parameter affected the result were reached. After selecting the best value of one parameter, we fixed it and started selecting the value of the next parameter. The variable parameters are further described below in the order in which they were selected.

5.2.1 KERNEL AND ITS PARAMETERS

In our experiments, we adopted the two most popular functions as kernels: dot product and Gaussian kernel as shown in expressions (1) and (3), respectively.

With dot product being used as the kernel, the best result was obtained at CER = 0.32. This indicates that the algorithm performs very poorly when the dot product is used as a kernel. For the Gaussian kernel, we had to vary its standard deviation, sigma σ. For small σ, there was a slow increase in correctly recognized attempts as the threshold decreased. With a strong increase in σ from the optimal value, there was a general degradation of the ROC curve, without any characteristic features; at further increase, the iteration algorithm stopped converging, which, most likely, was due to rounding errors inherent in calculations involving floating-point numbers. The optimal value of σ for the presented sample turned out to be 101.

5.2.2 DISTANCE FROM THE CLUSTER CENTER

As described earlier, there are two ways (simplified and iterative) to find η, which is the distance from the cluster center at which the degree of membership is assumed to be 0.5. When using the simplified method, no additional parameters are required. The best CER obtained using the simplified method was 0.187. For the iterative method, the outlier proportion must be specified. Experimentally, the best expected outlier proportion was found to be 0.1. Using the iterative algorithm, it gives a CER of 0.177. When the outlier proportion is varied between 0.05 and 0.2, the CER varies between 0.181 and 0.177.

5.2.3 DATA NORMALIZATION FACTORS

The most significant improvements in performance were obtained after normalizing the input data before processing. The reason for this is that the parameters, by their nature, have very different values. For example, the dwell time of a key is always strictly positive, while the interval between a key release and the next key press time may be positive or negative. Therefore, some normalization factors were considered, namely normalization to the square root of the variance, the absolute deviation, the square root of the absolute deviation, the interquartile range, and median absolute deviation. The absolute deviation and its square root gave the most accurate results for normalization.

A point that should be mentioned here is that for some users, the first type of normalization turned out to be better, and for others the second type. In this regard, we attempted to find out the best normalization method at the time of training. Cross validation was used for this purpose. The training sample was divided into two halves – the first part was normalized to the absolute deviation and to the root of the absolute deviation. Then two models were trained on it and tested on the second half of the training sample, respectively. After the results were obtained, training was similarly performed on the second half and testing on the first half. The best normalization method for the user was the one with the smaller mean CER. However, similarly, introduction of cross validation did not improve the CER.

Additionally, when processing data, we replaced each value x in the input data with natural logarithm *ln(x+C)*, where the value of *C* was taken as large as possible, such that *ln(x+C)* could be computed. This decision was justified by the fact that random variables describing individual password typing features, as was found out during the experiment, were more or less lognormally distributed.

6. RESULTS AND DISCUSSION

The chart in figure 2 compares the overall performance of some of the algorithms considered in [43] with the suggested algorithm. The algorithms have been rank-ordered in alphabetical order. In the chart, our proposed algorithm is designated as "SUGGESTED ALGORITHM".

The suggested algorithm obtained the lowest crossover error rate (0.093), thus indicating higher accuracy and reliability. The Manhattan (scaled), Nearest Neighbor (Mahalanobis), and the Outlier Count (z-score) detector were the other top-performing detectors using the crossover error performance measure. Our algorithm turned out to be 0.003 better than the best performer – the Manhattan (scaled), at 0.096.

The anomaly-detection algorithms were appraised based on the same data, under the same conditions, and using the same procedures. Therefore, differences in performance can be credited to the algorithm and not to different experimental conditions.

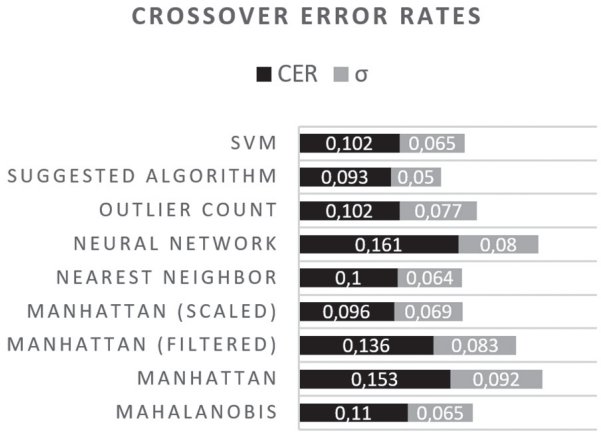

Fig. 2. A comparison of the performances of anomaly-detection algorithms

A critical challenge here is that minor differences in the algorithms and even in the assessment can trigger substantial changes in performance. Typing biometrics is a delicate instrument in a noisy domain. As long as assessment and comparison depend on controlling these small differences in performance, shared data and similar assessment procedures are crucial. So extra shared data and further assessments are required to determine and disentangle the factors facilitating and hindering the performance of each algorithm.

To further validate obtained results, the zero-miss false-alarm rate (ZM-FAR) could be computed and compared across the anomaly-detection algorithms considered. To calculate the ZM-FAR, the threshold is chosen such that FAR is minimized under the constraint that the miss rate be zero. This measure was used in some earlier studies [44, 45]. The CER and ZM-FAR are different performance measures, but both are error rates (i.e., lower values imply fewer errors and better performance).

7. CONCLUSION

In this work, we have investigated keystroke biometrics-based static authentication problem. In doing so, we proposed a new model – based on fuzzy sets and kernel function – for representing raw keystroke data, developed and implemented the corresponding algorithm, and compared it with some classic anomaly-detection algorithms, via experiments, on an equal basis. Our suggested method was found to have outperformed existing methods (with respect to the static authentication problem) – obtaining the lowest crossover error rate.

We have made some trade-offs, which certainly influenced performance. For this reason, the data could be used to assess what impact such decisions have. To give an example, we used 200 samples for training, which may seem unrealistically large. Moreover, we used unpracticed illegitimate users, which appears to be impractical because such users might practice if they knew timing mattered, thereby enhancing detector performance. We have made these trade-offs for the sake of unbiased assessments.

To achieve high performance with less training data, a different appraisal procedure could be adopted to train the detection algorithm using fewer passwords. However, such should be categorically and rigorously described to avoid conflating and confusing different appraisal methods.

8. REFERENCES

[1] W. H. Yang, S. P. Shieh, "Password authentication schemes with smart cards", Computers & Security, Vol. 18, No. 8, 1999, pp. 727–733.

[2] D. S. Carstens, P.R. McCauley-Bell, L.C. Malone, R.F. DeMara, "Evaluation of the human impact of password authentication practices on information security", Information Science Journal, Vol. 7, No. 1, 2004, pp. 67–85.

[3] I. Sluganovic, A. Karlovic, P. Bosilj, M. Šare, S. Horvat, "User authentication based on keystroke dynamics analysis", Proceedings of the 35th International Convention MIPRO, Opatija, Croatia, 21-25 May 2012, pp. 2136–2141.

[4] M. Styugin, "Dynamic key password authentication", International Journal of Security and Networks, Vol. 14, No. 2, 2019, pp. 78-85.

[5] G. Shafi, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks", SIAM Journal on Computing, Vol. 17, No. 2, 1988, pp. 281–308.

[6] U. Maurer, "Intrinsic limitations of digital signatures and how to cope with them", Proceedings of the 6th International Conference on Information Security, Bristol, UK, 1-3 October 2003, pp. 180–192.

[7] A. Levi, C. B. Güder, "Understanding the limitations of S/MIME digital signatures for e-mails: A GUI based approach", Computers and Security, Vol. 28, No. 3-4, 2009, pp. 105–120.

[8] RecFaces, "What is biometric security and why does it matter today?", https://recfaces.com/articles/biometric-security (accessed: 2021).

[9] M. Chrobok, Physical Biometrics vs Behavioral Biometrics, https://www.revelock.com/en/blog/physical-biometrics-vs-behavioral-biometrics (accessed: 2021).

[10] I. Alsaadi, "Study on most popular behavioral biometrics, advantages, disadvantages and recent applications: a review", International Journal of Scientific & Technology Research, Vol. 10, No. 1, 2021, pp. 15–21.

[11] M. Bača, M. Schatten, J. Ševa, "Behavioral and physical biometric characteristics modeling used for ITS security improvement", Transport Problems, Vol. 4, No. 4, 2009, pp. 5–13.

[12] T. Grance, M. Stevens, M. Myers, "Guide to Selecting Information Technology Security Products", NIST Guide to Selecting Information Technology Security Products (NIST Special Publication 800-36), 2003.

[13] P. S. Teh, A. B. Teoh, S. Yue, "A survey of keystroke dynamics biometrics", The Scientific World Journal, Vol. 2013, p. 408280, 2013.

[14] M. S. Obaidat, "Verification methodology for computer systems users", Proceedings of the 1995 ACM Symposium on Applied Computing, Nashville, Tennessee, USA, 26-28 February 1995, pp. 258–262.

[15] V. N. Gudivada, V. V. Raghavan, V. Govindaraju, C. R. Rao, "Handbook of Statistics", Cognitive Computing: Theory and Applications, Vol. 35, 2016, pp. 2–384.

[16] H. F. Tipton, M. Krause, "Information Security Management Handbook", Auerbach Publications, 6th Edition CRC Press LLC, 2007.

[17] E. Conrad, S. Misenar, J. Feldman, "CISSP Study Guide, Syngress", 2nd Edition, 2012, p. 599.

[18] E. Conrad, S. Misenar, J. Feldman, "Eleventh Hour CISSP: Study Guide", Syngress, 3rd Edition, 2017, p. 200.

[19] F. W. M. H. Wong, A. S. M. Supian, A. F. Ismail, "Enhanced user authentication through typing biometrics with artificial neural networks and K-nearest neighbor algorithm", Proceedings of the 35th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 4-7 November 2001, pp. 911–915.

[20] E. Lau, X. Liu, C. Xiao, X Yu, "Enhanced user authentication through keystroke biometrics", Computer and Network Security, Vol. 6, 2004.

[21] F. Bergadano, D. Gunetti, C. Picardi, "User Authentication through keystroke dynamics", ACM Transactions on Information and System Security, Vol. 5, No. 4, 2002, pp. 367–397.

[22] G. Saggio, G. Costantini, M. Todisco, "Cumulative and ratio time evaluations in keystroke dynamics to improve the password security mechanism", Journal of Computer and Information Technology, Vol. 1, No. 2, 2011, pp. 4–11.

[23] R. Solanki, P. Shukla, "Estimation of the user's emotional state by keystroke dynamics", International Journal of Computer Applications, Vol. 94, No. 13, 2014, pp. 21–23.

[24] K. S. Sung, S. Cho, "GA SVM wrapper ensemble for keystroke dynamics authentication", Proceedings of the International Conference on Biometrics, Hong Kong, China, 5-7 January 2006, pp. 654-660.

[25] M. I. Petrovsky, "Outlier detection algorithms in data mining systems", Programming and Computer Software, Vol. 29, No. 4, 2003, pp. 228–237.

[26] B. Schölkopf, A. J. Smola, "A short introduction to learning with kernels", Advanced Lectures on Machine Learning, Lecture Notes in Computer Science, Springer, 2003.

[27] T. Hofmann, B. Schölkopf, A. J. Smola, "Kernel methods in machine learning", The Annals of Statistics, Vol. 36, No. 3, 2008, pp. 1171–1220.

[28] Y. Cho, L.K. Saul, "Kernel methods for deep learning", Proceedings of the 22nd International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 7-10 December 2009, pp. 342-350.

[29] H. Chiroma, S. Abdulkareem, A.I. Abubakar, T. Herawan, "Kernel functions for the support vector machine: comparing performances on crude oil price data", Recent Advances on Soft Computing and Data Mining, Springer, 2014, pp. 273–281.

[30] C. Savas, F. Dovis, "The impact of different kernel functions on the performance of scintillation detection based on support vector machines", Sensors, Vol. 19, No. 23, 2019, p. 5219.

[31] M. A. M. Hasan, S. Xu, M. M. J. Kabir, S. Ahmad, "Performance evaluation of different kernels for support vector machine used in intrusion detection system", International Journal of Computer Networks and Communications, Vol. 8, No. 6, 2016, pp. 39–45.

[32] F. Meng, Y. Fu, F. Lou, Z. Chen, "An Effective Network Attack Detection Method Based on Kernel PCA and LSTM-RNN", Proceedings of the International Conference on Computer Systems, Electronics and Control, Dalian, China, 25-27 December 2017, pp. 568–572.

[33] Z. Rustam, N. Olievra, "Comparison of fuzzy robust Kernel C-Means and support vector machines for intrusion detection systems using modified kernel nearest neighbor feature selection", Proceedings of the 3rd International Symposium on Current Progress in Mathematics and Sciences, Bali, Indonesia, 26–27 July 2017.

[34] N. Baili, H. Frigui, "Fuzzy clustering with multiple kernels in feature space", Proceedings of the IEEE International Conference on Fuzzy Systems, Brisbane, QLD, Australia, 10-15 June 2012, pp. 1–8.

[35] L. A. Zadeh, "Fuzzy sets", Information and control, Vol. 8, No. 3, 1965, pp. 338–353.

[36] D. Klaua, “About an approach to multi-valued set theory”, Monatsblatt Deutscher Akademie der Wissenschaft, Berlin, 1965, pp. 859–876. (in German)

[37] C. Kahraman, B. Öztayşi, S. Çevik Onar, "A comprehensive literature review of 50 years of fuzzy set theory", International Journal of Computational Intelligence Systems, Vol. 9, Sup 1, 2016, pp. 3–24.

[38] Y. Nesterov, Efficiency of coordinate descent methods on huge-scale optimization problems, SIAM Journal on Optimization, Vol. 22, No. 2, 2010, pp. 341–362.

[39] P. Richtárik, M. Takáč, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function", Mathematical Programming, Series A, Vol. 144, No. 1-2, pp. 1–38.

[40] A. Ene, H. L. Nguyen, “Random coordinate descent methods for minimizing decomposable submodular functions”, Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6-11 July 2015, pp. 787–795.

[41] R. Wang, C. Chen, J. Lee, E. F. Darve. "PBBFMM3D: A parallel black-box algorithm for kernel matrix-vector multiplication", Journal of Parallel and Distributed Computing, Vol. 154, 2019, pp. 64–73

[42] R Core Team. “R: A language and environment for statistical computing. R Foundation for Statistical Computing”, Vienna, Austria. http://www.R-project.org/ (accessed: 2020)

[43] K. S. Killourhy, R. A. Maxion, "Comparing anomaly-detection algorithms for keystroke dynamics", Proceedings of the International Conference on Dependable Systems & Networks, Estoril, Lisbon, Portugal, 29 June - 2 July 2009, pp. 125-134.

[44] S. Cho, C. Han, D. H. Han, H. Kim, "Web-based keystroke dynamics identity verification using neural network", Journal of Organizational Computing and Electronic Commerce", Vol. 10. No. 4, 2000, pp. 295–307.

[45] E. Yu, S. Cho, "GA-SVM wrapper approach for feature subset selection in keystroke dynamics identity verification", Proceedings of the International Joint Conference on Neural Networks, Portland, OR, USA, 20-24 July 2003, pp. 2253–2257.