Keyboard Usage Authentication Using Time Series Analysis

Abdullah Alshehri(✉), Frans Coenen, and Danushka Bollegala

Department of Computer Science, The University of Liverpool, Liverpool L69 3BX, UK
{a.a.alshehri,coenen,danushka.bollegala}@liverpool.ac.uk

Abstract. In this paper, we introduce a new approach to recognising typing behaviour (biometrics) from an arbitrary text in heterogeneous environments using the context of time series analytics. Our proposed method differs from previous work directed at understanding typing behaviour, which was founded on the idea of usage a feature vector representation to construct user profiles. We represent keystroke features as sequencing discrete points of events that allow dynamically detection of suspicious behaviour over the temporal domain. The significance of the approach is in the context of typing authentication within open session environments, for example, identifying users in online assessments and examinations used in eLearning environments and MOOCs, which are becoming increasingly popular. The proposed representation outperforms the established feature vector approaches with a recorded accuracy of 98 %, compared to 83 %; a significant result that clearly indicates the advantage offered by the proposed time series representation.

Keywords: Keystroke recognition · Keystroke time series · Typing patterns

1 Introduction

Biometrics are acknowledged to provide a robust method for authenticating users based on their personal traits, as opposed to *token-based* mechanisms (such as passwords). Personal traits can be classified as being either behavioural or physiological [14]. The usage of behavioural biometrics has received prominent attention in the context of user authentication because they offer the advantage that they do not require specialised equipment [20]. Unlike physiological biometrics (for example fingerprints or iris data) that do require such equipment. One form of behavioural biometric is keystroke patterns; the typing patterns produced when an individual uses a keyboard. Keystroke patterns are a promised behavioural biometric that can provide unobtrusive authentication to confirm the legitimate users. A frequently cited example of the use of keystroke patterns for user authentication purposes is to confirm user credentials (such as password and username). In this case, authentication process is conducted by

S. Madria and T. Hara (Eds.): DaWaK 2016, LNCS 9829, pp. 239–252, 2016.
DOI: 10.1007/978-3-319-43946-4_16

comparing timing features of successive keystrokes with a stored typing profile so as to authenticate the person inputting the credentials [3,9,11,13,18]. To date keystroke patterns are typically represented as feature vectors comprised of quantitative statistical values, for instance, the calculated average flight time (interval) of frequent consecutive pairs of *graphs* (keypress sequences), usually *bi*-graphs [9]. A comparison, between a learnt user profile and previously unseen profiles, is then performed using a variety of paradigms, such as classification, AI based or Neural Network, to see whether two corresponding profiles are matched. However, the process is becoming harder in the case of dealing with arbitrary (free) text where constructing the feature vector is becoming stochastic. The reason is that the typed text is expected to be different each time; and therefore, the sequencing of key presses is largely lost. There is also a great deal of variability in the statistical features used to construct the feature vectors. Consequently, the reported results to date have tended not to be as good as anticipated to apply in heterogeneous environments [1,4,8,10,12,16]. The conjecture of studying typing patterns based on free text is that keystroke can be applied to continuous surveillance in heterogeneous environments where typing patterns are extracted from the arbitrary text. For example, it is sensible to be employed for continuous authentication in online assessments and examinations frequently used in eLearning environments and MOOCs[1], which is becoming increasingly popular.

The idea of this work is directed to deal with keystroke feature representation in the context of time series paradigm rather than using feature vectors based classification approach. The intuition is that time series representation can be more readily used to identify dynamically suspicious behaviours from free text. Furthermore, time series avails to capture keystroke sequences, unlike in the case of statistical techniques. We have considered that a typing session is represented as a series of discrete points P_M expressed in the temporal domain, where $_M$ is the number of points in a keystroke time series. Each point P is defined as pairs $P = (t, k)$, where t is a time stamp or time identifier; and k is depended multi-dimensional keystroke features. The diversity of keystroke timing features allow us to implement the proposed representation in two ways: (i) one in the 2D space as Flight time F^t (interval time between consecutive keystrokes) is recorded along the y-axis, where indexing keystrokes KN is along the x-axis; and (ii) representing features in 3D space (x,y,z) where we use F^t along the y-axis, Hold time HD^t (the length time of pressing a key) over the z dimension, and indexing keystrokes KN along the x-axis ticks. The purpose of implementing the two methods of representation (2D and 3D) is that to evaluate the effectiveness of using multivariate features for keystroke time series, and to compare which one can result in a more understanding of typing patterns in time series paradigm.

The main contribution of this work is that to introduce a different representation of keystroke timing features in the context of time series analysis to extract

[1] Massive Open Online Course (MOOC): is a web-based teaching distance that allows users to participating a variety of learning resources including filmed lectures, board discussion, etc. It is widely becoming used in the academic teaching process. See https://www.mooc-list.com/.

meaningful patterns in heterogeneous environments. Thus, keystroke biometrics can be eligible to use in different disciplines not only for authentication purposes, such as psychological detection [2], intrusion detection [17] or deceptive writing recognition [5].

The remainder of this paper is organised as follows. In Sect. 2, related work of keystroke feature representation methods is reviewed. Section 3 introduces a description of keystroke time series representation. Similarity method of keystroke time series is then discussed in Sect. 4. The evaluation and comparison of the proposed approach are reported on in Sect. 5. Finally, the work is summarised and concluded in Sect. 6.

2 Previous Work

There is a little work in the literature that has investigated the use of typing patterns generated from free text for user authentication purposes [1]. Most studies, as noted in the introduction to this paper, have adopted a feature vector representation where the features are computed statistical measurements. For example Dowland and Furnell [7] used digraph latency for the feature vector representation from which a binary classifier was generated. The classifier operated using the mean and standard deviation of digraph occurrences in a training profile. Principal disadvantages of the approach were that to achieve a reasonable classification performance a substantial amount of data was required with which to train the classifier. Furthermore, a dedicated classifier was necessary for each individual. The approach would thus be difficult to apply in heterogeneous environments such as eLearning platforms and MOOCs. An alternative approach was presented in Gunetti and Picardi [10] where the average time for pressing frequent sequences (n-graphs) was recorded and stored in arrays, one per n-graph. Common n-graphs were extracted for corresponding samples (reference and test). The elements of the arrays were then ordered and the distance between sample pairs computed by comparing the ordering in the reference array with the order in the test array. This measure was referred to as "the degree of disorder". However, learning a reference feature sample depends on all other samples in the reference profile. This can cause an efficiency issue when dealing with large numbers of samples as would be expected with respect to heterogeneous environments. Ahmed *et al.* [1] used key-down time information and the average digraph flight time to represent feature vectors to be employed in the context of a classifier. Although they obtained good results in heterogeneous environments, the issue is that the scalability of results is largely influenced by changing the environments conditions, such as using different keyboard layout. Indeed, the demand for developing such generic mechanism that able to recognise typing patterns in heterogeneous environments is desirable. Thus, the monitoring of keystroke sequencing over the temporal domain is argued for a better understanding of the arbitrary text, unlike constructing vectors to interpret the extracted features. The concept of using time series analysis, to the best knowledge of the authors, has not been considered in the previous work on keystroke free text detection.

3 Keystroke Time Series

Time series is a sequential ordering of data points that occur within an interval time [19], as each point corresponds multiple values. We first start with providing basic definitions in regards to keystroke time series:

Definition 1. *A Keystroke Time Series K_{ts}: is an ordered discrete sequence of points P; $K_{ts} = \left[P_1, P_2, \ldots, P_i, \ldots, P_M\right]$ where $M \in \mathbb{N}$ is the length of series and P_i is a tuple corresponding pairs of dimensional features.*

Thus, different keystroke time series may have different lengths M that describing an independent typing task in the session.

Definition 2. *A point tuple P_i in K_{ts}: is dependent dimensional features consists of two instances $\langle t, k\rangle$ where: (i) t is the indexing sequence of time stamp (KN) in which keys are pressed; and (ii) k is a set of timing attributes and descriptive features including: flight time (F^t), key-hold (KH^t) and key code (K_{cod}). So each p_i can be formally written as $p_i = \langle t_i, k_i\rangle$ where:*

- $\forall p_i \in K_{ts} : p_i \leftarrow \langle t_i, k_i\rangle$
- $\forall t_i \wedge k_i \in p_i : t_i = KN; k_i = \{F_i^t, KH_i^t, K_{cod_i}\}$

Definition 3. *Keystroke time series subset S: is a set of keystroke time series with length L, generated from K_{ts}, $S = \left[p_1, p_2, \ldots, p_i, \ldots, p_L\right]$ where $L < M$.*

Based on the above definitions, we can exploit the dimensionality of keystroke features to visualise time series in different spaces (2D and 3D). This is expected to give a better explanation of the typing rhythm of free text when adopting multi-features in the temporal domain to discriminate unique patterns.

3.1 2D Keystroke Time Series Representation

Keystroke temporal events have been represented as 2D series using two features: F^t and HD^t, respectively. The indexing sequence KN has been used along the x-axis, where F^t value or HD^t are used along the y-axis. Thus, a tuple p_i is underlying the sequential ID number KN per keystroke for t_i; and F^t or HD^t as for k_i, so a keystroke sequence can be simultaneously represented as $K_{ts} = \{\langle KN_1, F_1^t\rangle, \langle KN_2, F_2^t\rangle, \ldots\}$, $K_{ts} = \{\langle KN_1, HD_1^t\rangle, \langle KN_2, HD_2^t\rangle, \ldots\}$.

Recall that the value of F^t has to meet a pre-defined threshold value θ, to ensure the fluency of sequence; otherwise, some long stops over the typing session may affect similarity measurement (as we describe later on Sect. 4). We have considered the value of 3000(ms) as a normal variation, $\theta = 3000$. In Algorithm 1, if the value of F^t is greater than the threshold θ (line 7), then reduce F^t to Zero. So, every point with Zero value is considered as a reasonable stopping of typing. Figure 1 shows F^t values in one independent task that has been taken from our dataset. It can be observed in Fig. 1(a) that outlier values of F^t can describe a spurious behaviour where Fig. 1(b) depicts keystroke series after minimising F^t values.

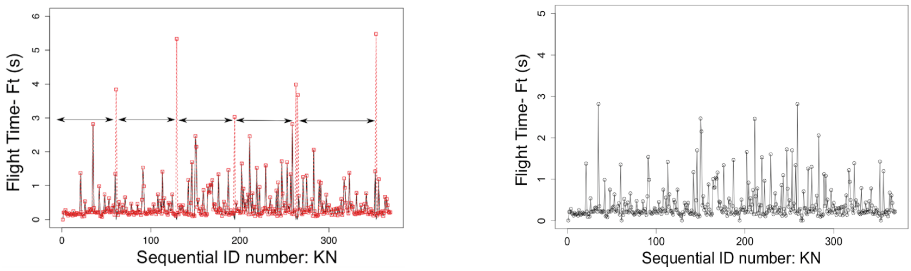


Fig. 1. Keystroke time series before removing outlier values of F^t **(a)**; and keytroke time series after removing outlier values of F^t **(b)**.

Thus, each typing session is represented as 2D time series that can discriminate a distinct pattern of typing. Figure 2(a) and (b) give two keystroke time series, taken from our evaluation dataset (see Sect. 5), for Subject 2 writing two different texts. It can be observed that this subject has a steady rhythm fluctuating between 0.1 and 1 ms. In contrast, Fig. 2(c) and (d) show two time series, for Subject 9, have a range between 0.1 and 1.5 with some peaks that favourably can introduce a similarity typing pattern for the same subject. From the figure, it can be seen that there are apparent dissimilarities in the keystroke pattern between the different subjects (despite writing different texts).

Algorithm 1. Removing Outlier Values of Flight Time (F^t)

Require: $K_{ts} \leftarrow$ keystroke time series, $\theta \leftarrow$ threshold outlier value.
Ensure: $\widehat{K_{ts}} \leftarrow$ Reduce outlier values in K_{ts}.

```
K_ts = (p_1, p_2, ..., p_i, ..., p_L)
p_i ← ⟨t_i, k_i⟩
L ← length of K_ts
for i = 1 to L do
    for each p_i in K_ts do
        p_i ← ⟨F_i^t⟩                    ▷ Search only for the value of F^t
        if p_i >θ: then
            p_i == 0
            K_ts^ = Update (K_ts)
        end if
    end for
end for
Return K_ts^
```

3.2 3D Keystroke Time Series Representation

Further dependent features have been employed to conceptualise keystroke time series in 3D representation. Representing keystroke time series in the 2D space,

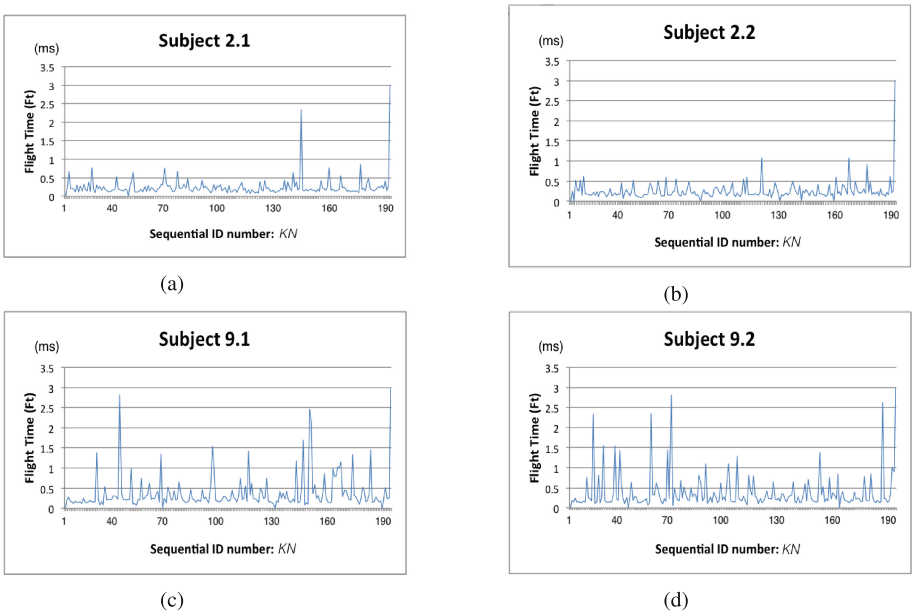


Fig. 2. Examples of keystroke time series representation: **(a)** and **(b)**, time series for Subject 2 writing two different texts; **(c)** and **(d)**, time series for Subject 9 writing two different texts.

in some cases, may affect the discrimination of patterns. For example, the time sequence may appear somewhat similar over some ticks in the series where it can influence the accuracy when calculating the similarity between two-time series. The conjuncture is then that 3D practically can show a preference to calculate the weight of series in three dimensions x, y, z rather than noticing data sequence in 2D space. To this end, we incorporated hold-time KH^t feature along the z-axis in the coordinate space. So, each point in the sequence p_i has: (i) the sequencing numbering KN over the x-axis, (ii) F^t over the y-axis, and (iii) HD^t over the z-axis. Thus, the tuple consists 3 dimensional features as $\forall p_i \in K_{ts} : p_i = \langle [t_i : KN], [k_i : F^t, KH^t] \rangle$.

4 Measuring the Similarity of Keystroke Time Series

Having represented keystroke time series, the similarity can be computed between the current series with one or more series. Given two keystroke time series $K_{ts1} = \{p_1, p_2, \ldots p_i, \ldots, p_M\}$ and $K_{kt2} = \{q_1, q_2, \ldots q_j, \ldots, q_N\}$, where M and N is the length of the two series, the simplest way to define similarity s is by directly computing the Euclidean distance between each points. However, this requires both time series to be of the same length $M = N$, where this is not necessarily the case at all time. The similarity should be performed between sequences that have varied lengths (when $M \neq N$). To this end, Dynamic Time

Warping (DTW) is the best choice that allows for non-linearity matching of two-time series with different lengths [15].

4.1 DTW Similarity in 2D Space

Lets assume that we have the above keystroke time series K_{ts1} and K_{ts2}, and also assumed that $M \neq N$. The two corresponding time series are constructed in a matrix X with $N \times M$, $X = (c_1, c_2, \ldots, c_n, \ldots, c_L)$, $N \leqslant L \leqslant M$. The elements of X are then computed by the squared distance D of F^t between the two corresponding points p_i and q_j.

$$c_{ij} \leftarrow D(c_n) = \sqrt{(F_j^t - F_i^t)^2} \tag{1}$$

The lowest cumulative distance ξ in each cell is the founded as in the following equation:

$$\xi(c_{ij}) = D + P_{min(\varepsilon(i,j-1),\, \varepsilon(i-1,j-1),\, \varepsilon(i-1,j))} \tag{2}$$

where D is the current distance of the i-th and j-th points in the cell c_{ij} and P is the lowest value obtained form: (i) the vertical cell $(i, j-1)$, (ii) diagonal cell $(i-1, j-1)$, and (iii) the horizontal cell $(i-1, j)$. The idea is then to find the lowest cost of the path (*Warping Distance* WD), where it is describing continuous cells in the matrix that mapping the alignment between K_{ts1} and K_{ts2}. The lower WD concerning the two time series being compared the similar the two time series are; if $WD = 0$, the two time series are identical.

$$WD = \min[\sum_{n=1}^{n} \xi(c_{ij})_n] \tag{3}$$

Recall that at the same manner, the value of WD is calculated when applying independently HD^t as the feature of interest in time series representation. Therefore, we compare the obtained values of WD for the both applied features, respectively, as described in Sect. 5.

4.2 DTW Similarity in 3D Space

With respect to 3D representation, we slightly modified the concept presented for 2D to perform DTW similarity. As described in Sect. 3.2, two features of interested, F^t and HD^t have represented two depended dimensions. This can affect measuring the distance for each cell in the matrix X. To avoid some computational conflicts, we simply find a weighted value w^* for each point p_i in the 3D space rather than separately computing the distance for each instances. The value of w^* is founded by computing the percentage between flight time F^t and hold time HD^t as in the following equation:

$$w_{p_i}^* = \sqrt{\log\left(\frac{F_i^t}{HD_i^t}\right)^2} \tag{4}$$

Therefore, the elements of matrix X is filled by calculating the distance between each corresponding $w^*_{q_j}$ and $w^*_{p_i}$ in the series.

$$c_{ij} \leftarrow D(c_n) = \sqrt{(w^*_{q_j} - w^*_{p_i})^2} \tag{5}$$

By constructing the matrix X, the WD is then computed at the same method described for 2D representation.

5 Evaluation

For the evaluation propose, we have examined the proposed method to detect typing patterns by simulating the operation of on-line assessments where students were asked to respond to discussion questions. A number of experiments have been conducted by applying the proposed representation in 2D and 3D; and then to compare the operation with statistical feature vector approach similar to that used in earlier work on free text typing recognition.

5.1 Data Collection

Keystroke timing data was collected (in milliseconds) using a Web-Based Keystroke Timestamp Recorder (WBKTR) developed by the authors. WBKTR was developed in JavaScript whereas it can work on cross-platforms web browsers. There is no need to install a third party or plug-ins, so it works smoothly without annoying users with further obligations. Another advantage of using JavaScript is that to avoid any implications for network delay when passing data to the server, which can affect the accuracy of recorded time. Thus, the script function works at the end user station to record time stamp within the current accuracy of the computer clock. Ideally, this can give a reliable accuracy of the recorded time. A front-end page in HTML was showing three discussion questions, similarly to the board discussion applied in eLearning environments. The interface can be found at (http://cgi.csc.liv.ac.uk/~hsaalshe/WBKTR3.html). A total of 17 subjects at the graduate level, ages between 20–35 were asked to response questions (for simplicity we used the term subject to refer each participant). The identity of the respondents was anonymised for privacy concerns. They were asked to type at least 100 words in response to each question with no maximum limitation so that adequate numbers of keystrokes (not less than 100 keystrokes per question) could be collected. For convenience, a scripting function was used to count the number of words per question. Samples with a number of keystrokes less than 100 (per question) were discarded. The reason is that 100 keystrokes can sufficiently provide a meaningful typing pattern [10]. Figure 3 illustrates that a total number of keystrokes more than 100 gives a steady accuracy of pattern detection. During the session, flight time F^t and hold time HD^t are recorded per each keystroke. A PHP script was used to store the identified attributes in the form of a plain text file on a server side for each subject.

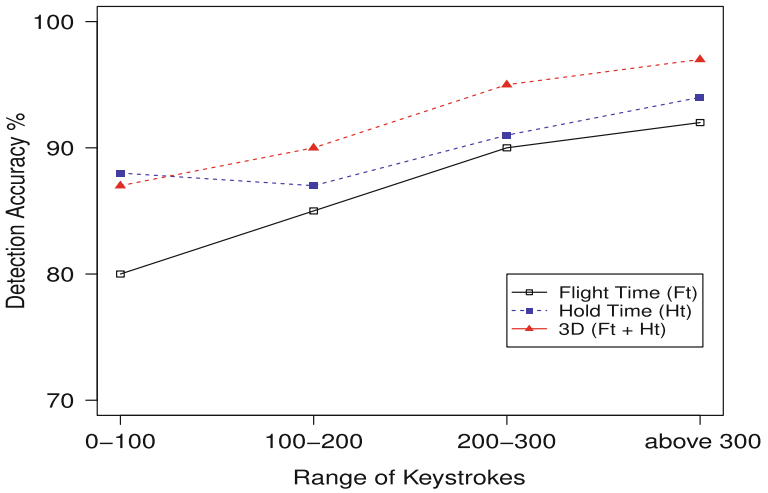


Fig. 3. The recorded accuracy with a range of keystroke number between 100 and 300 for different features in time series (Ft, Ht, 3D). The indication is that the detection accuracy is steadily performed while the number of keystrokes is above 100.

5.2 Analysis

The total number of samples (keystroke time series) N that we obtained is 17. Each sample S_i was splitted into three equal lengths of subset (keystroke subset time series) $\{s_{1i}, s_{2i}, s_{3i}\}$, as one for training and the other two for testing purpose. This results in a total number of $m = 3 \times N$ samples. This division allows to expand the comparison status by grouping samples into three main groups: (i) Group a, as $a = \{s_{11}, s_{12}, \ldots, s_{1i}\}$, (ii) Group b, as $b = \{s_{21}, s_{22}, \ldots, s_{2i}\}$; and (iii) Group c, as $c = \{s_{31}, s_{32}, \ldots, s_{3i}\}$. So, we implemented multiple experiments by swapping groups each time. This gave us also a wider comparisons each time as we simulated different (training and testing) samples for the same subject. The different groups of dataset being compared is then as follows: (i) $a. \vee \{b, c\}$, (ii) $b. \vee \{a, c\}$ and (iii) $c. \vee \{a, b\}$, the symbol $\vee$ is used to denote the versus status.

Figure 4 simplifies the idea of matching different groups. The warping distance WD is then performed as explained in Sect. 4. Figure 5(a) illustrates the WD of two samples s_{1i} and s_{2i} from the same user, whilst Fig. 5(b) shows the WD of two samples from two different users. A distinction can clearly be perceived. Lets perform the comparison in the combination group $a. \vee \{b, c\}$, for each sample s_{1i} the WD was compared with that of all the remaining samples $s_{\{2,3\}_i}$. A similarity threshold σ, for each subject, has been calculated by the average

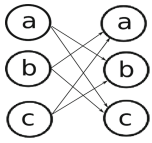
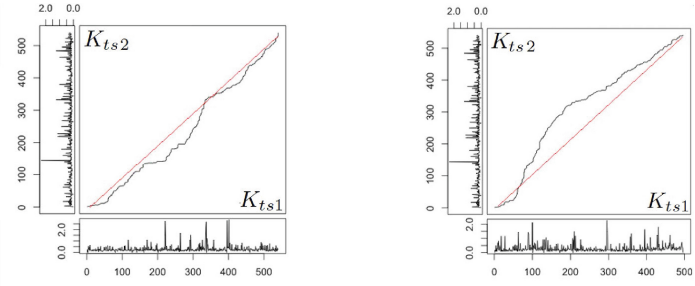


Fig. 4. Dataset has been divided into three groups a, b, c; and multiple comparisons have been conducted between different combination of groups: $a. \vee \{b, c\}$, $b. \vee \{a, c\}$, $c. \vee \{a, b\}$.



(a) WD of the same subject (b) WD for two different subjects.

Fig. 5. Application of DTW. It can be observed that the WD is more alignment in **(a)** as the sequencing of two time series for the same subject.

value of WD for s_{1i}, s_{2i} and s_{1i}, s_{3i}. Whenever a sample s_{1i} was found to be the most similar to σ this was considered to be a *correct* match; otherwise, the match was deemed to be *incorrect*. Thus, each smaple has a rank value r by ordering the WDs for each corresponding sample in the ascending order. The detection accuracy was computed as the ratio between the number of incorrect matches ℓ prior to a correct match being arrived at ($\ell = \sum(r-1)$) and the total number of test cases τ ($\tau = n \times (m-1)$). For simplicity, we calculate the accuracy of 3D representation as: $\frac{850-17}{850} \times 100 = 98.10\,\%$, as 17 is the value of ℓ and 850 is the value of τ (Table 3). As the same manner, we have implemented 2D representation using F^t and HD^t features, respectively. Tables 1, 2 and 3 introduce the accuracy results obtained from each feature with respect to time series representation.

False Rejection Rate (FRR) and False Acceptance Rate (FAR): As same as other biometric applications, we have evaluated our proposed approach by calculating the percentage of False Rejection Rate (FRR); and False Acceptance Rate (FAR). According to the European Standard for access control, the acceptable rate of FRR is 1 %, where the rate of FAR is 0.001 % [14]. Thus, we used these metrics to measure how far our proposed, as biometric authentication, from this standard.

In each combination in our experiments, we calculated FRR by computing the number of subjects n that their samples' rank r is not equal to 1, $\sum_1^n r \neq 1$. If the equivalent sample's rank are not equal to 1, this means that sample is falsely rejected. In contrast, FAR is calculated by the number of samples that recorded a higher rank than the current equivalent samples where all samples, smaller than the current equivalent sample, supposed to be accepted as real users.

Tables 1, 2 and 3 present the recorded results of FRR and FAR obtained from our proposed representation, comparing with feature vector representation as describe later in Sect. 5.3. We can clearly observe that time series representation recorded the best values for FRR and FAR in all different combinations of experiments.

5.3 Comparison with Feature Vector Approach

To obtain a reasonable evaluation of our proposed approach, we have examined the concept of the statistical feature vector style of operation found in earlier work; and compare the performance of the two methods of representations. This was performed by computing the average flight time $\mu(f^t)$ (Eq. 6) and hold time $\mu(h^t)$ (Eq. 7), respectively, for the most frequently occurring di-graphs found in the dataset.

$$\mu(f^t) = \frac{1}{n}\sum_{i=1}^{i=n} Ft_n \tag{6}$$

$$\mu(h^t) = \frac{1}{n}\sum_{i=1}^{i=n} HD_n^t \tag{7}$$

where n is the number of identified frequent di-graphs. In this manner feature vectors could be generated for each sample. The resulting representation was thus similar to that found in more traditional approaches to free text recognition [1,7,10,16]. Each sample S_i is divided into three vectors v_{1i}, v_{2i}, v_{3i}. As the same scenario in time series representation (Subsect. 5.2), we have measured the similarity between two vectors using Cosine Similarity (CS). Thus, for two vectors v_{1i} and v_{2i}, CS calculated as:

$$CS(v_{1i}, v_{2i}) = \frac{v_{1i} \cdot v_{2i}}{||v_{1i}|| \times ||v_{2i}||} \tag{8}$$

where $v_{1i} \cdot v_{2i}$ is the dot product between two feature vectors v_{1i} and v_{2i}, and $||v_{1i}||$ ($||v_{2i}||$) is the magnitude of the vector v_{1i} (y). In the same manner, as described above, we measured the similarity of each feature vector with every other feature vector using CS. Note that using CS, unlike in the case of DTW, the feature vectors need to be of the same length. In this case, the results for each subject are listed in descending order of CS ($CS = 1$ indicates a perfect match). We also computed FRR and FAR for the feature vector as well. The results obtained are presented in Tables 1, 2, and 3. It can be observed that the accuracy, in all combinations datasets, has recorded fewer values than our proposed method. It can be also noticed from the tables that a worse performance has been recorded than when using time series approach with respect to FRR and FAR.

An alternative evaluation measure that can be used to indicate the effectiveness of the proposed approach to keystroke time series is Mean Reciprocal Rank (MRR) [6]; a measure that indicates how close the position of a desired subject of interest is to the top of a ranked list. MRR is a standard evaluation measure used in Information Retrieval (IR). MRR is calculated as follows:

$$MRR = \frac{1}{|Q|}.\sum_{i=1}^{|Q|}\frac{1}{r_i} \tag{9}$$

where: (i) Q is a set of queries (in our case queries as to whether we have the correct subject or not), and (ii) r_i is the generated rank of the desired response to Q_i.

Table 1. Results obtained by applying F^t as the feature applied for the two methods of representation, *Time Series*, and *Feature Vector*.

Representation	*2D Time Series with* F^t				*Feature Vector with* F^t			
Metrics / Dataset	FRR(%)	FAR(%)	MRR	Acc(%)	FRR(%)	FAR(%)	MRR	Acc(%)
$a. \vee \{b, c\}$	6.11	1.52	0.438	93.88	20.58	1.64	0.283	79.41
$b. \vee \{a, c\}$	**5.17**	1.41	**0.520**	**94.82**	21.64	1.76	0.155	78.35
$c. \vee \{a, b\}$	6.70	**1.17**	0.454	93.29	19.17	1.64	0.225	80.82

Table 2. Results obtained by applying HD^t as the feature applied for the two methods of representation, *Time Series*, and *Feature Vector*.

Representation	*2D Time Series with* HD^t				*Feature Vector with* HD^t			
Metrics / Dataset	FRR(%)	FAR(%)	MRR	Acc(%)	FRR(%)	FAR(%)	MRR	Acc(%)
$a. \vee \{b, c\}$	**2.70**	1.05	0.658	**97.29**	20.35	1.64	0.199	79.64
$b. \vee \{a, c\}$	3.64	0.94	0.666	96.35	16	1.64	0.251	84
$c. \vee \{a, b\}$	3.52	**0.70**	**0.723**	96.47	17.17	1.64	0.242	82.82

Thus, with reference to Table 3, time series representation has recorded the best values of MRR with comparing with feature vector representation. Among different methods of time series representation, 3D representation has outperformed other features with a value of 0.801 while the best MRR value on feature vector is = 0.311 (Table 3).

Table 3. Results obtained by applying F^t and HD^t as features applied for the two methods of representation, (i) *Time Series*, and (ii) *Feature Vector*.

Representation	*3D Time Series*				*Feature Vector with* F^t *and* HD^t			
Metrics / Dataset	FRR(%)	FAR(%)	MRR	Acc(%)	FRR(%)	FAR(%)	MRR	Acc(%)
$a. \vee \{b, c\}$	2	0.70	0.745	98	15.29	1.52	0.305	84.70
$b. \vee \{a, c\}$	**1.76**	**0.58**	**0.801**	**98.20**	17.80	1.50	0.311	82.11
$c. \vee \{a, b\}$	1.80	0.70	0.772	98.10	17.50	1.60	0.275	82.40

For completeness, the average value has been computed for each feature applied in all combinations for the both methods of representation. Table 4 summarises the average values obtained for time series representation and feature vectors in all metrics. It can be clearly observed that 3D method outperforms other representations, including 2D keystroke time series method. A clear indicator that applying multi-variate time series has promising potential to detect typing patterns from arbitrary text.

Table 4. Summary of the average values obtained for all representations.

Representation	*Time Series*				*Feature Vector*			
Metrics / Applied Feature	FRR(%)	FAR(%)	MRR	Acc(%)	FRR(%)	FAR(%)	MRR	Acc(%)
Average results for F^t	5.99	1.37	0.470	94.00	20.46	1.68	0.225	79.53
Average results for HD^t	3.29	0.90	0.480	96.70	17.84	1.64	0.230	82.09
Average results for $3D$	**1.85**	**0.66**	**0.770**	**98.10**	16.86	1.54	0.210	83.07

6 Conclusion

An approach to recognise typing patterns in heterogeneous environments, that deal with arbitrary text of typing, has been proposed. The process operates by representing keystroke timing features as discrete points in a time series where each point has a timestamp of some kind and attribute value. The proposed representation used a sequential key-press numbering system in 2D, by applying flight time and hold time, respectively; and using both features in the 3D time series representation. DTW has been adopted to measure the similarity of keystroke time series so that practically works with non-linearity time series. By implementing the proposed approach to detect typing patterns in a simulated onLine environment, recorded results show that proposed feature representation obtained an overall accuracy of 98.10 % (coped with FRR = 1.85 %, and FAR = 0.66 %). This compared very favourably with the alternative approach using feature vector with an accuracy of 83.07 % (FRR = 16.86 % and FAR = 1.54 %) when applying classical features vector representation; a clear indication that the proposed time series based approach outperforms the vector based approach. The result demonstrated that the proposed time series based approach to keystroke authentication has a significant potential benefit in the context of user authentication in heterogeneous environments such as those used in online learning and MOOCS. The authors believe that further improvement can be realised by considering different methods of representation, such as Fast Fourier Transform (FFT). Future work will also be directed at confirming the findings using larger datasets.

Acknowledgment. We would like to express our thanks to those who participated in collecting the data and to Laureate Online Education b.v. for their support.

References

1. Ahmed, A.A., Traore, I.: Biometric recognition based on free-text keystroke dynamics. IEEE Trans. Cybern. **44**(4), 458–472 (2014)
2. Bixler, R., D'Mello, S.: Detecting boredom and engagement during writing with keystroke analysis, task appraisals, and stable traits. In: Proceedings of the International Conference on Intelligent User Interfaces, pp. 225–234. ACM (2013)
3. Bleha, S., Slivinsky, C., Hussien, B.: Computer-access security systems using keystroke dynamics. IEEE Trans. Pattern Anal. Mach. Intell. **12**(12), 1217–1222 (1990)

4. Bours, P.: Continuous keystroke dynamics: a different perspective towards biometric evaluation. Inf. Secur. Tech. Rep. **17**(1), 36–43 (2012)
5. Choi, Y.: Keystroke patterns as prosody in digital writings: a case study with deceptive reviews and essays. In: Empirical Methods on Natural Language Processing (EMNLP) (2014)
6. Craswell, N.: Mean reciprocal rank. In: Liu, L., Özsu, M.T. (eds.) Encyclopedia of Database Systems, p. 1703. Springer, US (2009)
7. Dowland, P.S., Furnell, S.M.: A long-term trial of keystroke profiling using digraph, trigraph and keyword latencies. In: Deswarte, Y., Cuppens, F., Jajodia, S., Wang, L. (eds.) Security and Protection in Information Processing Systems. IFIP, vol. 147, pp. 275–289. Springer, US (2004)
8. de Lima e Silva Filho, S.R., Roisenberg, M.: Continuous authentication by keystroke dynamics using committee machines. In: Mehrotra, S., Zeng, D.D., Chen, H., Thuraisingham, B., Wang, F.-Y. (eds.) ISI 2006. LNCS, vol. 3975, pp. 686–687. Springer, Heidelberg (2006)
9. Stockton Gaines, R., William Lisowski, S., Press, J., Shapiro, N.: Authentication by keystroke timing: Some preliminary results. Technical report, DTIC Document (1980)
10. Gunetti, D., Picardi, C.: Keystroke analysis of free text. ACM Trans. Inf. Syst. Secur. (TISSEC) **8**(3), 312–347 (2005)
11. Joyce, R., Gupta, G.: Identity authentication based on keystroke latencies. Commun. ACM **33**(2), 168–176 (1990)
12. Messerman, A., Mustafic, T., Camtepe, S.A., Albayrak, S.: Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics. In: International Joint Conference on Biometrics (IJCB), pp. 1–8. IEEE (2011)
13. Ogihara, A., Matsumuar, H., Shiozaki, A.: Biometric verification using keystroke motion and key press timing for atm user authentication. In: International Symposium on Intelligent Signal Processing and Communications, ISPACS 2006, pp. 223–226. IEEE (2006)
14. Polemi, D.: Biometric techniques: review and evaluation of biometric techniques for identification and authentication, including an appraisal of the areas where they are most applicable. Reported prepared for the European Commision DG XIIIC 4 (1997)
15. Rabiner, L., Juang, B.-H.: Fundamentals of speech recognition. Prentice Hall (1993)
16. Shepherd, S.J.: Continuous authentication by analysis of keyboard typing characteristics. In: European Convention on Security and Detection, pp. 111–114. IET (1995)
17. Sridhar, M., Abraham, T., Rebello, J., D'souza, W., D'Souza, A.: Intrusion detection using keystroke dynamics. In: Das, V.V. (ed.) Proceedings of the Third International Conference on Trends in Information, Telecommunication and Computing. LNEE, vol. 150, pp. 137–144. Springer, Heidelberg (2013)
18. Syed, Z., Banerjee, S., Cukic, B.: Normalizing variations in feature vector structure in keystroke dynamics authentication systems. Softw. Qual. J. **24**(1), 137–157 (2014)
19. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.: Experimental comparison of representation methods and distance measures for time series data. Data Min. Knowl. Discov. **26**(2), 275–309 (2013)
20. Yampolskiy, R.V., Govindaraju, V.: Behavioural biometrics: a survey and classification. Int. J. Biom. **1**(1), 81–113 (2008)