Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information)
15/2 (2022), 81-91. DOI: http://dx.doi.org/10.21609/jiki.v15i2.1015

Bimodal Keystroke Dynamics Based Authentication for Mobile Application Using Anagram

Prasti Eko Yunanto[1], Ari Moesriami Barmawi[2]

[1,2]Intelligent Systems Research Group, School of Computing, Telkom University,
Jl. Telekomunikasi No. 1 Terusan Buah Batu, Bandung, 40257, Indonesia

Email:gppras@telkomuniversity.ac.id[1], mbarmawi@melsa.net.id[2]

Abstract

Currently, most of the smartphones recognize uses based on static biometrics, such as face and fingerprint. However, those traits were vulnerable against spoofing attack. For overcoming this problem, dynamic biometrics like the keystroke and gaze are introduced since it is more resistant against spoofing attack. This research focuses on keystroke dynamics for strengthening the user recognition system against spoofing attacks. For recognizing a user, the user keystrokes feature used in the login process is compared with keystroke features stored in the keystroke features database. For evaluating the accuracy of the proposed system, words generated based on the Indonesian anagram are used. Furthermore, for conducting the experiment, 34 participants were asked to type a set of words using the smartphone keyboard. Then, each user's keystroke is recorded. The keystroke dynamic feature consists of latency and digraph which are extracted from the record. According to the experiment result, the error of the proposed method is decreased by 23.075% of EER with FAR and FRR are decreased by 16.381% and 10.41% respectively, compared with Kim's method. It means that the proposed method is successful increase the biometrics performance by reducing the error rates.

Keywords: *biometrics, keystroke dynamic, identification, authentication*

1. Introduction

Due to the development of communications, information exchange is frequently used for business, education, etc. Therefore, security is important and necessary, especially for securing personal identity. Biometrics appears as an alternative to change the conventional technologies, such as passwords, and PIN that should be remembered and could be stolen[1].

Biometrics is a technology for personal identification based on physical and behavioral characteristics [1, 2]. Thus, it is not necessary to remember and it can not be stolen[1–3]. The fingerprint and face are the biometrics traits that are widely used on smartphones, because they provide high security and user convenience [4, 5]. However, those traits are vulnerable against spoofing attack, because it is easy to imitate those static features [6, 7]. Therefore, recently dynamic biometrics based on behaviors, such as gesture, gaze, keystroke, voiceprint, and touch screen pattern are frequently used. Kim et al.[8] states that keystroke dynamics authentication for the smart device, such as smartphone and tablet is a critical issue, such that it is necessary for future investigation.

According to Anil K. Jain's explanation in his book entitled 'Handbook of Biometrics' [1], there are two processes that can use biometrics, identification, and authentication/verification. The identification process is used for identifying the user's identity who owns the biometric, and authentication is used to check the validity of the user who owns the related biometric. In this study, biometrics is used in smartphones, because biometrics is suitable for the authentication process. The authentication process is needed to secure data stored on smartphones against attacks of theft, disclosure, destruction, loss, or data alteration because authentication ensures that only authorized users can access data in a smartphone [9].

This study used keystroke dynamic for user au-

thentication because no additional device is necessary to be used as a sensor [10]. Generally, the keystroke features are obtained by recording the time it takes from pressing the first key until releasing it (dwell time) and the time it takes from pressing the first key until releasing the second key (flight time)[11]. If the device is a touchscreen, then information of finger pressures and position can be used to enrich the features.

There are many issues that have been addressed in keystroke dynamics area, such as devices for data acquisition, feature representations, classification methods, and experimental/evaluation procedures [11]. Methods to obtain the features become a major challenge in keystroke dynamic biometrics. Bergadano et al.[10] proposed the features based on the ordered latency of n-graphs, while the similarity of two samples is calculated by the displacement of n-graphs. The result is that the error FAR is 0.01% and FRR is 4%. Kang et al.[12] proposed a keystroke dynamic using a long and free text string, and the features are obtained by grouping the digraphs into eight groups. Each group represents the combination of three disjoint areas on desktop keyboards (left, right, and space area). The problem of Kang's method is that the word used for authentication should contain fixed number of features that can be represented by eight R, L, and S pairs (where R is right, L is left, and S is space area). Thus, words used for authentication could not be randomly chosen. Kim et al.[8] improves Kang's features by grouping the ordered digraph latency, such that the number of features can be adjusted based on the number of words used as sample of each group. Based on the experiment's result, the error rate of Kim's method [8] is less than the error rate of Kang's one [12]. However, since Kim's method [8] only concern about latency or typing speed, the error is still 0.44. To increase the performance of keystroke dynamic recognition, features consists of digraph and latency feature are introduced.

The user identification is decided based on the scores between the order of n-graphs[10] and latency features extraction[8]. The motivation is to take advantage of each approach's strengths, especially for keystroke dynamic recognition using the touchscreen keyboard by combining these approaches, the biometrics performance can be increased. Furthermore, this study only observes the digraph, because the text used in this study is short, less or equal than five words. The feature scaling is also applied to increase the performance of the biometrics system. Based on the experiment's result, it is shown that error rate of the proposed method is less than the existing method.

In this study, the authentication system based on biometrics was evaluated based on the value of False Acceptance Rate (FAR) (also called False Positive Rate (FPR)) and False Rejection Rate (FRR) or also called False Negative Rate (FNR), where the system error rate in recognizing valid and unauthorized users is measured based on False Positive and False Negative [13].

Next, this paper is organized as follows. Section 1 discusses the background of the research, while section 2 describes the state of the art and related research, which is the basis for conducting this research. Section 3 describes the method proposed in this study, section 4 is an experiment and discussion of the experiment's result, and section 5 discusses the conclusions.

2. Keystroke Dynamics of Biometrics

According to the survey that have been conducted by Teh et al.[11], the keystroke dynamics consists of two types of information, labels and time (they are usually known as dwell and flight time). The label of a keystroke is used to construct the n-graphs, while the latency features of n-graphs is provided by time information of the keystroke. The digraphs (n-graphs with $n = 2$) are obtained by taking two consecutive characters, starting with each character in the text, while the trigraphs ($n = 3$) uses three consecutive characters [10]. For example, text $abcd$ can be arranged into digraphs ab 23, bc 17, and cd 21, where ab is label of the digraphs and 23 is duration for typing ab.

The latency feature is constructed using four variations of dwell and flight time [8, 12, 14] (see Figure 1): (a) ***du*** or **down-up**, the time it takes from pressing the first key until releasing it; (b) ***dd*** or **down-down**, the time it takes from pressing the first key until pressing the second key; (c) ***ud*** or **up-down**, the time it takes from releasing the first key until pressing the second key; while (d) ***uu*** or **up-up**, the time it takes from releasing the first key until releasing the second key. Furthermore, various features are extracted based on these four variations of dwell and flight time [8, 10–12, 14, 15].

2.1. Keystroke Dynamics Using User-adaptive Feature Extraction

Based on Kim et al.[8], many keystroke dynamics focus on the fixed and short size of text. Since, typing fixed and short text such as password and token is not enough to guarantee user validity once the user login[8, 10], then to resolve this problems, the long and free typed text is used for keystroke

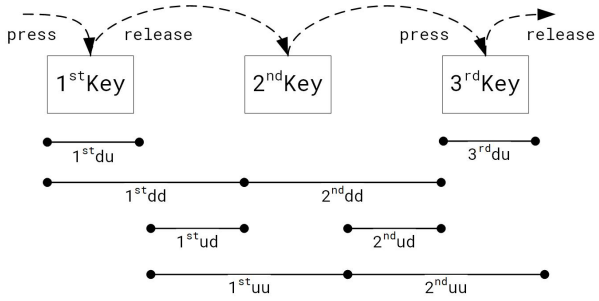

Figure 1. Variation of Dwell and Flight time of a single key.

dynamics. For data acquisition, 150 participants provided around 13,000 keystrokes based on Korean and English language. From 13,000 keystrokes, 5,000 keystrokes were used for training the model and 8,000 keystrokes are used to evaluate the model. Besides, to compare the authentication performance, samples with size of 100, 300, 500, 700, and 1,000 are chosen from the keystrokes used to train and test the system in order to provide the features.

The main challenge of this study is to find effective feature extraction methods for the long and free typed text. Kim et al.[8] proposed a feature extraction method focused on the typing speed, where the ordered latency such as duration of digraphs, dd, du, ud, uu are grouped, where each group forms one feature vector. Each vector is created by calculating the average of the digraphs' latency for the corresponding groups. It means the digraph which has similar latency will be grouped into the same group. If there are N groups, then these groups represent an N- dimensional feature vector.

Furthermore, for verifying each feature vector one of the following algorithm is used: (a) Gaussian density estimation. This algorithm uses the assumption that the dataset is generated from a single Gaussian normal distribution, where the mean and covariance of the training dataset are used as the parameters. If Gaussian probability is significantly high, then the user is considered as a legitimate user. Otherwise, it is recognized as an impostor; (b) Parzen window density. This algorithm uses the assumption that its data has Gaussian distribution and probability. The Gaussian distribution and probability are obtained by calculating the average of each distribution probability; (c) One-class SVM. This method is used to find the hyperplane for separating the valid user and the impostor; (d) k-NN. This algorithm uses k nearest distance of samples to verify whether a user is a valid user or not; and (e) k-means clustering. This algorithm uses the nearest centroid of training data for verifying, whether a user is a valid user or not. Finally, it can be concluded that the recognition result for each verification algorithm is provided based on the geometric average of the five feature vectors.

Based on Kim's experiment [8], the improvement of the average of error rates (EER) and standard deviation is about 0.05 and 0.1 respectively, compared with Kang's method [12], where the lowest error rates of 1,000 samples for each train and test data using Kim's method [8] is about 0.44.

3. Implementation of Biometrics System Based On Keystroke Dynamics

This section discusses the detail of biometrics system based on keystroke dynamics. The keystroke dynamic based authentication system was divided into three main processes: (a) registration; (b) login; and (c) authentication.

The keystroke was enrolled using an Android-based data acquisition application that had to be installed on the participant's smartphone. The basic idea of keystroke based on authentication system was conducting authentication based on the keystroke based feature. The feature is generated using the data acquisition process (as discussed in section 3.2.1) and feature extraction process (as discussed in section 3.2.2).

The user's keystroke feature consisted of the digraph and its latency (the digraphs and the latency is already discussed in section 2) which were used in registration and login process. The registration process was used to collect the keystroke feature and store it into the features database. The login process was conducted to obtain the keystroke feature for authentication process. The authentication process was conducted to authenticate whether a user was a legitimate user or not by comparing between the score of the keystroke features obtained from registration and the login process.

The overview of the keystroke biometrics based on authentication system is shown in Figure 2.

3.1. Authentication Scheme

Authentication scheme was conducted to evaluate whether the user is the legitimate one or not. This process was conducted by comparing between the user's keystroke feature obtained from login process and user's keystroke feature stored in the database.

In the real implementation, the authentication scheme was conducted as follows:

1) **Registration**
   a) User registration begins by typing some text based on the text generated by the system.

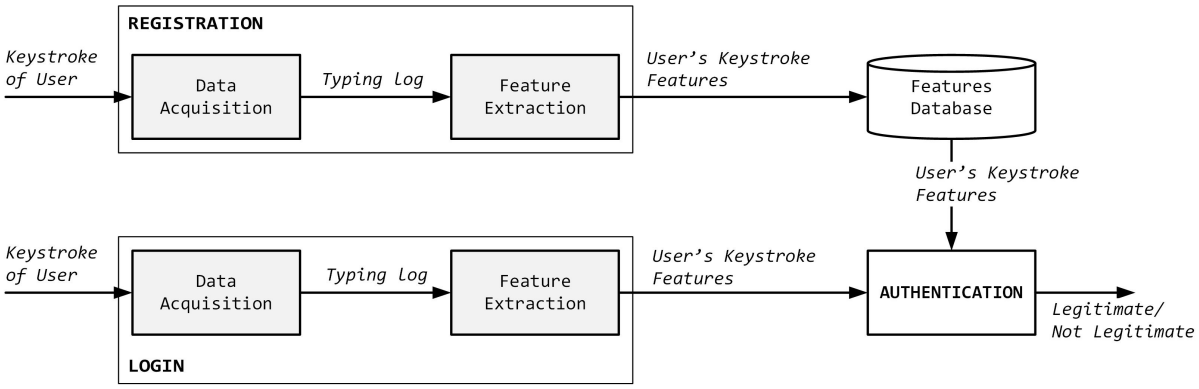

Figure 2. The overview of keystroke biometrics based on authentication system.

The keystroke resulting while typing the text will generate the keystroke features. These features were stored in the database.

b) Calculating the similarity of the keystroke features resulted from step 1a.

c) Finding the optimal threshold based on the similarity obtained from step 1b. The optimal threshold was obtained if FAR and FRR were equal. The threshold was further stored in the database.

2) **Login**

a) The user was typing some text based on the text generated by the system. The user's keystrokes during the typing process were stored temporarily. Then, the system generated the keystroke features based on the obtained keystroke.

b) Calculating the similarity between the keystroke features obtained from step 2a of the login process and the keystroke features stored in the database.

3) **Authentication**

a) The similarity resulting from step 2b of the login process was compared with the threshold resulting from step 1c of the registration process.

b) If the similarity resulting from step 2b of the login process is greater than or equal to the threshold, then the user is identified as legitimate, otherwise the user is identified as illegitimate.

c) After the authentication process is successful, the user can access the intended data.

3.2. Registration

The registration process was used to collect the keystroke feature template of the user. The registration process consisted data acquisition and feature extraction process. Data acquisition process was used to obtain the typing log, while the feature extraction process was used to obtain user's keystroke feature template.

3.2.1. Data Acquisition. In the registration process, the data acquisition process was conducted as follow:

1) The application generate a text that had to be typed by the user.
2) The user had to type the text using the keyboard provided by the application.
3) The characters that had been typed (keystroke) and the time for typing them (timestamp) were recorded in the typing log.
4) The timestamp consisted of the dwell and flight time.

The output of the data acquisition process was the typing log consisted of keystrokes and timestamp.

In this study, nine types of the Indonesian anagram were used, where each word in the anagram had five variations of word and around 225 characters in total. The acquisition was conducted by 45 repetitions of typing process, with one day of an interval between acquisitions. In this case, 34 participants were invoked to obtained keystrokes using an Android-based application. The simulation was conducted by typing the combination of each anagram appeared on the screen.

The user interface of the data acquisition application is shown in Figure 3.

Due to key-events for any key on soft input method (an on-screen keyboard) was not supported by Android SDK then customed QWERTY keyboard was used [1].

[1]https://developer.android.com/training/keyboard-input/commands.html#SingleKey

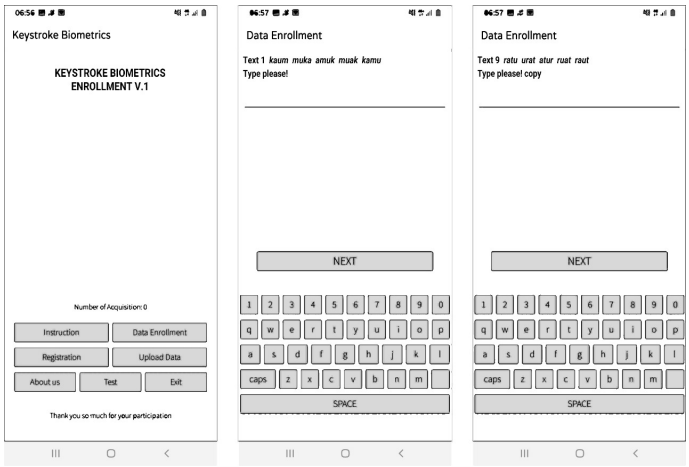

Figure 3. Screen layout for data acquisition.

Data acquisition process consists of two processes: (1) generating the text; and (2) typing the text.

1) **Generating The Text**
Generating the text process was used to generate the text that had to be typed by a user. The text was chosen from several combinations of the anagram. For instance, the word *"amuk"* can be generated into *"kamu"*, *"kaum"*, *"muak"*, and *"muka"*. There were nine variations of anagram text that were generated, where each word used five variations of text (see Table 1).

Table 1. Words of Indonesian anagram.

No	Original word	Anagram word			
1	**amuk**	kamu	kaum	muak	muka
2	**asri**	rasi	rias	sari	siar
3	**karam**	karma	makar	marak	marka
4	**karet**	raket	rekat	retak	terak
5	**kista**	sakit	sakti	sikat	taksi
6	**kasur**	rakus	rasuk	rusak	sukar
7	**paus**	puas	sapu	suap	usap
8	**kuil**	kuli	liku	liuk	ulik
9	**atur**	ratu	raut	ruat	urat

2) **Typing The Text**
Typing the text process was used to collect the user's keystroke during typing, where the application records a log that contained the timestamp of the pressed/released key. The typing log was represented by 3-tuple (P, G_{tx}, T_{set}), where P is the user ID, G_{tx} is the generated text, and the set of tuple that represented as $T_{set} = \{(k_i, pt_i, rt_i) \mid 1 \leq i \leq N, \text{and } N \in \mathbb{N}\}$, where for each i-th keystrokes, k represents the key's label that are typed, pt and rt are the time when a key is pressed and released respectively, while N is the number of keystrokes.

3.2.2. Feature Extraction. Feature extraction process was conducted to extract T_{set} from the typing log and generate user's keystroke feature. The user's keystroke feature consisted of two types of feature, the ordered digraph and the normalized latency feature.

For obtaining these features, there were two sub-processes that should be conducted: (1) digraph construction; and (2) feature modeling, as shown in Figure 4.

1) **Digraph Construction**
Digraph construction process was conducted for obtaining the digraphs and its latency obtained from the typing log. A digraph is a combination of two-letters or keys that have an adjacency sequence within text. Let a text S consists of character $s_1, s_2, \ldots, s_N$, then the digraphs d of text S can be obtained using $d_{igraph} = \{(s_i, s_{i+1}) \mid 1 \leq i \leq N-1, \text{and } N \in \mathbb{N}\}$, where s_i is the i-th character in text S and N is number of character in text S. Figure 5 shows the digraphs of "*Hello World*".

Let T_{set} be a set of tuple obtained from the typing log, then the detail of digraphs construction process is explained in the following procedure:

a) Building the digraphs using d_{igraph} by arranging the set of tuple T_{set}, where each $i-th$ digraph d_i is represented as $(k_i, k_{(i+1)})$, where $1 \leq i \leq M$ and M is number of possible digraphs.
b) Calculating the latency of each digraph d_i $(lt_i = (dd_i, du_i, ud_i, uu_i))$ from tp and tr

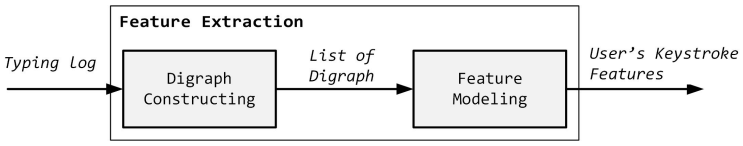

Figure 4. Block diagram of feature extraction process.

S : "Hello World"
$\boldsymbol{d}_{igraphs}$: (H, e),(e, l),(l, l),(l, o),(o, SPACE),(SPACE, W),(W, o), (o, r),(r, l), and (l, d)

Figure 5. Digraph of "*Hello World*".

of T_{set} using Equation 1, 2, 3, and 4.

$$dd_i = tp_{i+1} - tp_i \quad (1)$$
$$ud_i = tp_{i+1} - tr_i \quad (2)$$
$$uu_i = tr_{i+1} - tr_i \quad (3)$$
$$du_i = tr_i - tp_i \quad (4)$$

Constructing the list of digraph D_{list} by pairing the digraph d and latency lt, as $D_{list} = \{(d_i, lt_i) \mid 1 \leq i \leq M, \text{and } M \in \mathbb{N}\}$,where M is the number of possible digraphs.

2) **Feature Modeling**
After obtaining the list of digraph, then the list of digraph was extracted into the keystroke features consisted of the ordered digraph and the normalized latency. There were three subprocesses that should be conducted in this modeling process: (a) merging the duplicate digraphs; (b) sorting the latency; (c) normalizing the latency; and (d) representing the general features.
   a) **Merging the Duplicate Digraphs**
   The process for merging the duplicate digraphs was conducted for removing the duplicated digraphs by merging the duplicated digraph into a single digraph including its latency. The merged latency was obtained by calculating the average of the duplicated digraph latency. Supose $D_{list} = \{(d_i, lt_i) \mid 1 \leq i \leq M \text{and } M \in \mathbb{N}\}$. If $d_i = d_j$ for $i \neq j$ then for merging the duplicated digraphs of d_i in the list of digraph D_{list} into a single digraph, Algorithm 2 should be conducted. The merged digraph $(d_i^{'})$ and its latency $(lt_i^{'})$ denoted by $D_{list}^{'} = \{(d_i^{'}, lt_i^{'}) \mid 1 \leq i \leq M^{'} \text{and } M^{'} \in \mathbb{N}\}$, where $M^{'}$ is number of digraphs after merging process.
   b) **Sorting the Latency**
   The process for sorting the latency were conducted to obtain the list of digraph and latency in ascending order according to its latency. Since the latency consisted of four elements (dd , ud, du, and uu) then the digraph were sorted based on each element of the latency. Does, the result of sorting the latency process is $D_{dd}^{'}$,$D_{ud}^{'}$,$D_{du}^{'}$, and $D_{uu}^{'}$, where $D_k^{'} = \{(d_i, lt_i^k) \mid 1 \leq i \leq M^{'} \text{and } M^{'} \in \mathbb{N}\}$ and k is dd, ud, du, or uu.
   c) **Normalizing the Latency**
   This process was conducted for obtaining the feature vector based on the latency. The process consisted of three subprocesses, namely grouping the latency based on each element, normalizing the latency by calculating the average of all latencies in the group, and scaling the latency using unit vector (see Equation 8).
   This process for grouping the ordered latency into $N^{'}$ group features is explained as follows:
      i) Calculating the division of number of digraphs $M^{'}$ and the number of features $N^{'}$. The result of the division was consisted of two variables d and r, where d is an integer result of the division, and r is the remainder.
      ii) Calculating the feature vector element f_{ki} using Equation 5.

$$f_{k_i} = \begin{cases} \dfrac{\sum_{j=1}^{d} k^{'}_{((i-1)\times d)+j}}{d} & r = 0 \\ \dfrac{\sum_{j=1}^{d+y} k^{'}_{((i-d)\times d)+x+j}}{d+y} & r \neq 0 \end{cases} \quad (5)$$

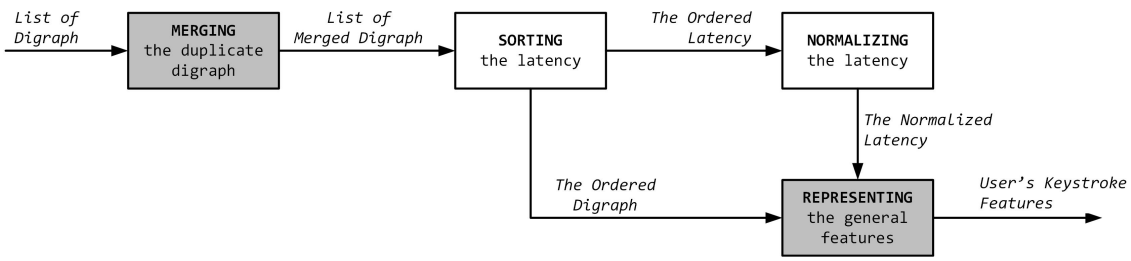

Figure 6. Block diagram of feature modeling.

Algorithm 1. Find Unique Digraph.

```
procedure findUniqueD(D_list, U)
  input  D_list = {(d_i, lt_i)_{i=1}^{M}}
  output U = {(d_i)_{i=1}^{M'}}
begin
  U = empty
  U.M' = 0
  for i = 1 to D_list.M do
    if D_list.d_i not in U then
      U.add(D_list.d_i)
      U.M' = U.M' + 1
    endif
  endfor
endprocedure
```

Algorithm 2. For Merging Digraph.

```
procedure merging(D_list, D'_list)
  input  D_list = {(d_i, lt_i)_{i=1}^{M}}
  output D'_list = {(d'_i, lt'_i)_{i=1}^{M'}}
begin
  K = {dd, ud, du, uu}
  findUnique(D_list, D)
  for d' in D do
    temp_k = 0    {for ∀k ∧ k ∈ K}
    j = 0
    for i = 0 to D_list.M do
      if D_list.d_i == d' then
        temp_k = temp_k + D_list.d_i.k
        j = j + 1
      endif
    endfor
    lt'.k = temp_k / j
    D'_list.add((d', lt'))
  endfor
endprocedure
```

Where k is dd, ud, du, or uu, while x and y are as follows:

$$x = \begin{cases} i-1 & i \leq r \\ r & i > r \end{cases} \quad (6)$$

$$y = \begin{cases} 1 & i \leq r \\ 0 & i > r \end{cases} \quad (7)$$

iii) Constructing the feature vector $F_k = (f_{k1}, f_{k2}, \ldots f_{kN'})$

iv) Scaling the feature vector using unit vector. The unit vector is calculated using Equation 8.

$$\vec{x'} = \frac{\vec{x}}{\|\vec{x}\|} \quad (8)$$

Where $\vec{x'}$ is the unit vector of feature vector $\vec{x}$ and $\|\vec{x}\|$ is the Euclidean norm or resultant of vector $\vec{x}$.

d) **Representing The General Features**
This process was conducted for representing the user's keystroke feature. Each user's keystroke feature consisted of ($D^{'}_{dd}$,$D^{'}_{ud}$,$D^{'}_{du}$,$D^{'}_{uu}$) and (F_{dd},F_{ud},F_{du},F_{uu}).

3.3. Login

Login process was conducted to obtain the user's keystroke feature which is used for authentication. In the login process, user have to input the its user ID and conducted the processes for obtaining the data acquisition and feature extraction.

3.4. Authentication

The authentication process was conducted by calculating the similarity between user's keystroke features obtained from login process and user's keystroke feature template from the database. Two types of similarity were used: (a) The similarity

according to Euclidean distance as discussed in subsection 3.4.1; and (b) The similarity according to sequence of digraph as discussed in subsection 3.4.2. The similarity final score was calculated based on the average of those similarities.

3.4.1. The Similarity According to Euclidean Distance. This process was conducted for finding the feature vector similarity between two user's keystroke features. The detail of the process is explained as follows:

1) Calculating Euclidean distance between feature vector of user's keystroke features obtained from login process and user's keystroke features from the database using Equation 9.

$$Euc_k(F'_k, F_k) = \sqrt{\sum_{i=1}^{N'}(f'_k - f_i)^2} \tag{9}$$

Where Euc_k is the Euclidean distance between user's keystroke features from login process (F'_k) and user's keystroke features from database (F_k), while $k \in \{dd, ud, du, uu\}$ and N' is number of features.

2) Calculating the similarity using Equation 10.

$$Sf_k(F'_k, F_k) = 1 - Euc_k(F'_k, F_k) \tag{10}$$

Where Sf_k is the similarity between user's keystroke features from login process (F'_k) and user's keystroke features from database (F_k).

3) Calculating the similarity score F_{score} by calculating the geometric mean of Sf_k using Equation 11.

$$F_{score} = \sqrt{\prod_{k\in\{dd,ud,du,uu\}} Sf_k} \tag{11}$$

3.4.2. The Similarity According to Sequence of Digraph. This process was conducted to obtain the sequence of digraph similarity between user's keystroke features from login (D'_k) and user's keystroke features from database (D''_k) for each $k \in \{dd, ud, du, uu\}$. The detail of the process is explained as follows:

1) Finding the intersection digraph between the sequence digraph from login process and database. Let $I_k = d'_k \bigcap d''_k$ be an intersection digraph, where d'_k and d''_k are digraphs set of D'_k and D''_k respectively, then $I_k = \{(x_i, y_i) \mid 1 \leq i \leq N''\text{and } N'' \in \mathbb{N}\}$ where x_i and y_i are the characters or symbols of $i-th$ digraph, and N'' is the number of intersection digraphs.
2) Calculating the similarity of the sequence digraph between digraph obtained from login process and the database which obtained intersection digraph I_k. The similarity is calculated using Equation 12.

$$Sd_k = \frac{\sum_{i=2}^{N''} 1 - z_i}{\alpha} \tag{12}$$

Where Sd_k is the similarity of the sequence digraph, z_i is the absolute difference of digraph (x_i, y_i) between d'_k and d''_k, and $\alpha = \frac{(N'')^2}{2}$ for N'' is even or $\alpha = \frac{(N'')^2-1}{2}$ for N'' is odd. The absolute difference z_i is calculated using $z_i = \|j - i\|$, where $(x_i, y_i) = (x_j, y_j)$, $(x_i, y_i) \in d'_k$ and $(x_j, y_j) \in d''_k$.

3) Calculating the similarity score S_{score} by calculating the geometric mean of Sd_k using Equation 13.

$$S_{score} = \sqrt{\prod_{k\in\{dd,ud,du,uu\}} Sd_k} \tag{13}$$

Finally, the similarity between the two user's keystroke features was calculated using Equation 14.

$$Sim = \frac{F_{score} + S_{score}}{2} \tag{14}$$

The legitimation of the user was decided based on the similarity threshold, if the similarity score was greater than the threshold, then the user was recognized as the legitimate one and vice versa. The threshold was obtained by finding the equal error rates between FAR and FRR. The method for obtaining the threshold is discussed in section 4.1.

4. Result and Discussion

This section discusses about the measurement concept used for evaluating the performance of the biometrics based authentication system, the experiment and its result.

Performance of the system was calculated using error rates of identification, where the lower error rates indicated the better performance of the biometrics system. Three metrics were used for calculating the performance, namely EER, FAR, dan FRR [1].

4.1. Measurement Used in The Experiments

The performance of the biometrics based authentication was measured by calculating the Equal Error Rates (EER), when False Acceptance Rate (FAR) is equal to False Rejection Rates (FRR) [1, 8]. Since the error indicates the number of illegitimate users percentage which is recognized as a legitimate one (FAR) and vice versa (FRR) [1], then the error rates should be as lower as possible.

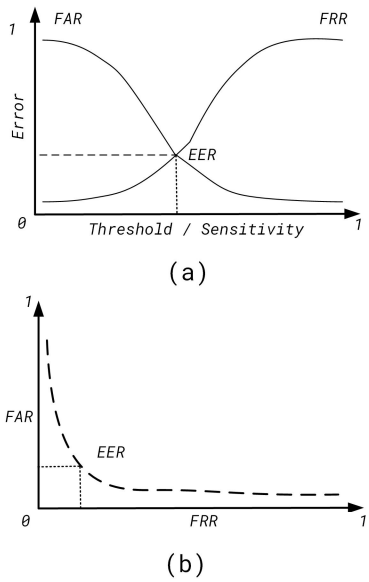

Figure 7. Relation between EER, FAR, and FRR using (a) Sensitivity curve and (b) Receiver Operating Characteristic (ROC) Curve [8].

Figure 7 shows that if the threshold is zero, which means that all illegitimate user will be recognized as a legitimate one, while if the threshold is higher than zero, then the legitimate user may be rejected. This condition was occurred because the higher similarity threshold, the higher number of illegitimate users which is recognized as a legitimate one and vice versa.

The similarity threshold is the boundary value where two user's keystroke features are said to be similar. The similarity is calculated by calculating the Euclidean distance between those two user's keystroke features (keystroke feature obtained from the database and login process).

$$FAR = \frac{f_A}{n_A} \quad (15)$$

$$FRR = \frac{f_R}{n_R} \quad (16)$$

Where f_A is the number of illegitimate users which is recognized as legitimate one and n_A is the number of illegitimate users, while f_R is the number of legitimate users which are recognized as the illegitimate one and n_R is the total number of legitimate users.

4.2. The Experiments

The experiment was conducted to find the optimal threshold of similarity. The optimal threshold was obtained when FAR and FRR were equal or balanced. Then, the threshold was tested using the new data to see the performance of the system to authenticate the legitimate user. The conducted experiment consisted of two scenarios.

1) The first scenario was used to find the optimal similarity threshold to separate the legitimate and not legitimate users in balanced error.
   In this scenario, the similarity between the keystroke features obtained from the registration process and the ones obtained from the login process was calculated. The FAR and FRR of the keystrokes obtained from the login process were calculated based on the chosen threshold. For finding a condition where FAR and FRR were equal, it is necessary to observe the FAR and FRR using different threshold values.
2) The second scenario was used for finding the similarity between the keystroke feature used in the login process and the one stored in the database.
   In this scenario, the feature of the keystroke entered by a user in the login process was used for finding the similarity between the keystroke feature used in the login process and the one stored in the database. A user is authenticated as a legitimate user if the similarity score is greater than or equal to the threshold.

The data used in the experiment were divided into three categories: (a) data for registration process; (b) data 2 for login process in the first scenario; and (c) data 3 for login process in the second scenario. In the experiments the performance of features extraction is compared with Kang's [12] and Kim's methods [8]. The number of keystrokes used in the experiment was 344,250

These keystrokes consist of 137,700 keystrokes used as the input of the registrations process, 206,550 keystrokes used as the input of the login process. The keystrokes used in the login process was divided into 137,700 used as data 1 for first experiment scenario and 68,850 keystrokes as data 2 for second experiment scenario the system.

4.2.1. The Experiment Result. This subsection discusses the result of the experiment scenario. First scenario for finding the similarity threshold that can separate the legitimate and not legitimate user in balanced error, and second scenario for finding the error of system using the threshold obtained from scenario 1.

Figure 8 shows the result of the first experiment scenario (EER) of Kang's [12], Kim's [8] and the proposed method. Based on the experiment re-

sult, Kang's method [12] achieved the EER average 0.319 (with a threshold of 0.778), Kim's method [8] achieved 0.304 (with a threshold of 0.880), and the proposed method achieved 0.234 (with a threshold of 0.641). It is shown that proposed method has the lowest error rates. Thus, the number of misidentification events is getting smaller.

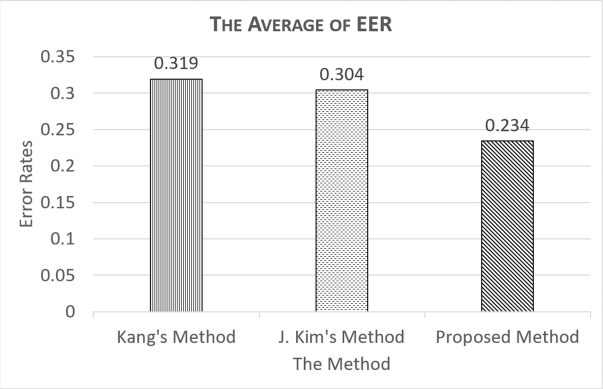

Figure 8. The comparison between EER of Kang's [12], Kim's [8] and the proposed method.

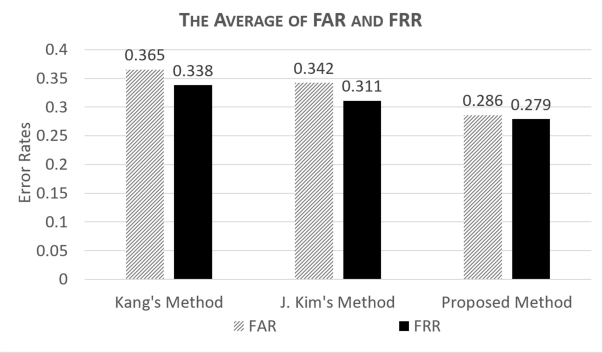

Figure 9. The comparison between FAR and FRR of Kang's [12], Kim's [8] and the proposed method.

Figure 9 shows the comparison between FAR and FRR in the second experiment scenario, using Kang's [12], Kim's [8], and the proposed method. It is shown that the average of FAR and FRR achieved by Kang's method [12] are 0.365 and 0.34, while Kim's method [8] achieved 0.342 and 0.311, and the proposed is 0.286 and 0.279. It is shown that proposed method has the lowest FAR and FRR. The FAR of the proposed method is 16.381 % less than the Kim's method [8], and the FRR is 10.410 % less than the Kim's method [8].

Compared with Kang's method [12], the error are reduced by 21.695% and 17.593% of FAR and FRR respectively, and Kim's method [8] by 16.381% and 10.410% of FAR and FRR.

Furthermore, the comparison between the error rates between FAR and FRR of the test result and the obtained ERR of the proposed method (see Figure 9) are increased. Kang's method [12] error is increased by 14.367% and 5.987% of FAR dan FRR respectively, and Kim's method [8] is decreased by 12.320% and 2.241% of FAR dan FRR respectively, while the proposed method is increased by 22.092% and 19.073% of FAR and FRR.

4.2.2. Discussion. According to Figure 8, it is shown that the EER of the proposed method is less than Kang's [12] dan Kim's methods [8]. Kim's method [8] has decreased the error rate of 5% compared with the Kang's method [12], while the proposed method has decreased the error rate of 23.074% compared with the Kim's method [8]. This condition was occurred because of the used of different devices, merging the same digraphs into one digraph, and introducing the digraph verification [10] in the proposed method, in this case, digraph was used instead of only using the latency.

Since, Kang's [12] and Kim's [8] methods were especially developed for computer desktop keyboard, then the obtained features from the smartphone were not relevant, due to the typing behavior using computer keyboard, and the smartphone was different. Although Kang [12] stated that the experiment has been conducted using mobile and touchscreen device, the three disjoint area of the hand in the keyboards can be effectively used for the user that uses all of their finger during typing words. Whereas on the smartphone, the user uses only single thumbs or both of them, such that the typing behavior when using computer is different with the smartphone devices.

Kim's [8] feature extraction can improve Kang's method [12], but Kim's method [8] only focuses on the typing speed, such that the latency feature of same digraphs may appeared in more than one group. As the consequence, the feature is not relevant because the same digraphs have to be in the same group. For example, the digraph x-y appears more than once in the text that has to be typed. In this case, actually the latency of those digraphs should be similar, and they are grouped in the same group, but since there is possibility that the latency of those digraphs are not similar, then there is possibility that the digraphs is grouped into the different group.

Therefore, merging the duplicated digraph was introduced in the proposed method, such that no possibility that the same digraph was grouped into different groups. The merging process can extract more relevant features, such that the error rate can

be decreased.

The digraph label [10] is introduced for improving the similarity evaluation. Since there is possibility that more than one digraph have similar latencies, then each digraph has to be labeled such that the order of the digraph can be verified. In this case, the order of digraph can be used for verifying the feature.

5. Conclusion

Recently, static biometrics such as fingerprint, face, etc. are frequently used for user authentication, however the features of static biometrics are easy to duplicate. Therefore, dynamic biometrics features, such as keystrokes, gesture, gaze, etc. is introduced for user authentication. In this study keystrokes dynamic is used to authenticate users of smartphones. The results of the experiments conducted shows that this approach was able to improve the performance of the keystroke-based biometrics system proposed by Kim et al. [8], because the EER value was reduced by 23.075% and FAR and FRR values can also be reduced by 16.381% and 10.41% respectively. This shows that the use of merging the duplicate digraphs and digraph verification can improve the system performance compared with the authentication system that uses the latency only.

References

[1] A. K. Jain, P. Flynn, and A. A. Ross, *Handbook of Biometrics*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[2] A. K. Jain, K. Nandakumar, and A. Ross, "50 years of biometric research: Accomplishments, challenges, and opportunities," *Pattern recognition letters*, vol. 79, pp. 80–105, 2016.

[3] A. Kołakowska, "Usefulness of keystroke dynamics features in user authentication and emotion recognition," in *Human-Computer Systems Interaction*. Springer, 2018, pp. 42–52.

[4] T. Chugh and A. K. Jain, "Fingerprint presentation attack detection: Generalization and efficiency," in *2019 International Conference on Biometrics (ICB)*. IEEE, 2019, pp. 1–8.

[5] V. Soum, S. Park, A. I. Brilian, Y. Kim, M. Y. Ryu, T. Brazell, F. J. Burpo, K. K. Parker, O.-S. Kwon, and K. Shin, "Inkjet-printed carbon nanotubes for fabricating a spoof fingerprint on paper," *ACS omega*, vol. 4, no. 5, pp. 8626–8631, 2019.

[6] S. Zhang, X. Wang, A. Liu, C. Zhao, J. Wan, S. Escalera, H. Shi, Z. Wang, and S. Z. Li, "A dataset and benchmark for large-scale multi-modal face anti-spoofing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 919–928.

[7] K. Cao and A. K. Jain, "Hacking mobile phones using 2D printed fingerprints," *Michigan State University, Tech. Rep. MSU-CSE-16-2*, 2016.

[8] J. Kim, H. Kim, and P. Kang, "Keystroke dynamics-based user authentication using freely typed text based on user-adaptive feature extraction and novelty detection," *Applied Soft Computing*, vol. 62, pp. 1077–1087, 2018.

[9] N. Zabidi, N. Norowi, and R. Wirza, "A survey of user preferences on biometric authentication for smartphones," vol. 7, 10 2018, p. 491.

[10] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, no. 4, pp. 367–397, 2002.

[11] P. S. Teh, A. B. J. Teoh, and S. Yue, "A survey of keystroke dynamics biometrics," *The Scientific World Journal*, vol. 2013, 2013.

[12] P. Kang and S. Cho, "Keystroke dynamics-based user authentication using long and free text strings from various input devices," *Information Sciences*, vol. 308, pp. 72–93, 2015.

[13] Y. Hong and R. Kumar, "Performance evaluation metrics for biometrics-based authentication systems," 2021.

[14] M. Antal, L. Z. Szabó, and I. László, "Keystroke dynamics on android platform," *Procedia Technology*, vol. 19, pp. 820–826, 2015.

[15] P. R. Dholi and K. P. Chaudhari, "Typing pattern recognition using keystroke dynamics," in *International Conference on Advances in Information Technology and Mobile Communication*. Springer, 2012, pp. 275–280.