Performance Enhancement on Keystroke Dynamics by Using Fusion Rules

Pin Shen Teh[1], Andrew Beng Jin Teoh[2], Thian Song Ong[1], Connie Tee[1]

***Abstract*— Keystroke dynamics refers to the timing information that expresses precisely when each key was pressed and released as a person types. In this paper, we present a novel keystroke dynamic recognition system by using a novel fusion approach. Firstly, we extract four types of keystroke latency as the feature data from our dataset. We then calculate their mean and standard deviation to be stored as template. The test feature data will be transformed into similarity scores via Gaussian Probability Density Function (GPD). We also propose a new technique, known as Direction Similarity Measure (DSM), to measure the trend differential among each digraph in a phrase. Lastly, various fusion rules are applied to improve the final result by fusing the scores produced by GPD and DSM. Best result with equal error rate of 2.791% is obtained when the AND voting rule is used.**

***Index Terms*—dwell time, flight time, keystroke dynamics, keystroke latency.**

I. INTRODUCTION

Our society today depends heavily on computers; they are fundamental parts of nearly every aspect of our lives. Security is a vital component of most computer systems, especially in E-Commerce activities over the internet. The most well known and familiar mechanism used to guarantee the security of the information system is through user authentication by textual passwords and PINs.

However, this kind of security method is weak in several aspects. Users tend to have the habit of writing down passwords and leave them in front of the computer or places which are highly exposed to the public. Besides, simple passwords are often easy to guess by using either dictionary or brute force attacks. In addition, negligent users normally use fragile passwords commonly composed of date of birth, phone numbers, nick names, vehicle registration number, and other clues which are easy to guess. All these aspects lead to compromise of user passwords without knowledge of their disclosure.

Biometrics is a unique authentication method which identifies a person based on his/her physiological or behavioral characteristics. A physiological characteristic is a relatively stable physical feature owned by a person, such as palm print or finger images, facial characteristics, and iris pattern. Behavioral characteristics are traits that are learned or acquired like signature, voice, gait, and keystroke dynamics. Keystroke dynamics is a behavioral biometric identification method based on the assumption that individual type in unique manners. Keystroke dynamics is an intelligent data processing technique which investigates the way a user types at a terminal by monitoring the keyboard inputs with an attempt to identify the user based on his/her habitual typing patterns. Keystroke dynamics functions in conjunction with conventional password authentication to provide extra layer of security to computer systems. It is non-intrusive and operates in parallel with the user's normal activities. Due to the wide social acceptance of the existing password measure, keystroke dynamics biometric system is readily to be accepted by the public. Apart from that, no additional hardware is required as this technology uses the existing keyboard and hence it is comparatively cheaper as compared to the other biometric systems.

In this paper, the main objective is to present and analyze a fusion approach which combines the score from a new yet simple method, namely Direction Similarity Measure (DSM), with the score generated by Gaussian Probability Density Function (GPD) on keystroke dynamics. This paper is organized as follows. A review of the previous work is presented in Section II while Section III introduces the methodology and algorithm used in this study. Section IV presents our proposed fusion approach and explain the fusion rules employed in the experiment. The results are reported and discussed in Section V. Finally, Section VI presents and discusses the conclusions and future work.

II. RELATED WORKS

As early as the 1860's, telegraph was the main method for long-distance communication. Telegraph operators recognized each other by the different ways they used to tap the telegraph keys. Today, the telegraph keys have been replaced by keyboard. It has been well-known that keyboard typing patterns of different users are unique among large population and the distinctive typing dynamics holds promise as individual identifier [1].

In year 1980, the first research on keystroke dynamics had been conducted by Gaines et al. [2]. The experiment involved 6 secretaries as subjects. Each was asked to type three passages, consisting 300-400 words each, at two different sessions. The keystroke interval was recorded for the experiment, and T-test was used to check whether two populations have the same average and standard deviation. Although the study achieved noteworthy result, 0% False Acceptance Rate (FAR) and 4% False Rejection Rate (FRR), it was impractical in real cases because of the low

[1] Faculty of Information Science and Technology, Multimedia University,
Jalan Ayer Keroh Lama, Bukit Beruang, 75450 Melaka, Malaysia,
psteh@mmu.edu.my, tsong@mmu.edu.my, tee.connie@mmu.edu.my
[2] Electrical and Electronic Engineering Department, Yonsei University, Seoul, South Korea, bjteoh@yonsei.ac.kr

number of users involved and the length of text used.

Five years later, Umphress and Williams [3] performed an experiment on 17 computer programmers. Each of them were required to type one passage of 1400 words which served as template, and another passage of 300 words as verification attempt. They identified the user by comparing the mean and standard deviation of the keystroke latencies and digraph between the reference profile and test data. Modest result of 17% FAR and 30% FRR were obtained. The major limitation of their experiment was that large typing string was required for generating reference profile and testing data.

In 1990, Joyce and Gupta [4] reported promising approach with 0.25% FAR and 16.67% FRR. The authors built a mean reference signature for eight sets of users' keystroke patterns consisting of username, password, first name, and last name. They then computed the norm of the test keystroke pattern to the mean reference signature, which was used to determine if a user was legitimate based on a predefined threshold.

Bleha and Obaidat [5] used Linear Perceptron as their classifier to verify the identity of users. The time durations between keystroke of user's password was collected. The data was collected from 10 valid users and 14 invalid users throughout a period of eight weeks. Half of the sample data collected was used as training data and the remaining half for testing. FAR of 8% and FRR of 9% was achieved using the proposed method.

A neural network approach was employed in year 2000 by Cho et al. [6]. Users typed their passwords for 150 to 400 times within several days. The last 75 timing vectors were separated for testing purpose. Keystroke duration time and interval time were used as the feature data in the experiment. The timing vectors which were determined as outliers were discarded. Result of 0% FAR and 1% FRR was reported. However, the research had some limitations. Firstly it was time consuming to train the model, secondly the data were preprocessed subjectively by human, and thirdly, a large data set was required to train the network.

While a lot of research work for keystroke dynamics focused on text phrase of various lengths, [7] performed their research on a six digits pin code typed on a numeric-pad. Their experiment involved 14 people. 50 samples of users and 13 random chosen imposters were used for training purpose. A multi-layer perceptron with the back-propagation learning rule were used. The system was able to achieve 9.9% FAR and 30% FRR. The main problem of using neural networks was that it required imposters' samples for training. Besides, the results suggested that the approach is not feasible for large scale use.

In [8], the training and testing data were collected from 43 users who type fixed string of length 37 for nine consecutive times over a period of two months. Monte Carlo approach was used to generate random simulated data from the users' samples. Approximately 19 times of the simulated data were generated to complement the 387 vectors of raw data. Four subsets were obtained from the training set which comprised of raw and simulated data. Wavelet transforms were performed to obtain a total of eight training subsets for each user. Parallel decision trees were used to authenticate users based on their keystroke patterns. A remarkable result of 0.88% for FAR and 9.62% for FRR was archived in the research.

Rodrigues et al. [9] implemented Hidden Markov Models (HMM) as classifier in their research. They limited their investigation over numerical passwords of length eight. Twenty people were invited to contribute to this experiment. Each individual was instructed to type their passwords ten times in four different sessions, yielding a total of 800 samples. An EER of 3.6% was obtained.

Hosseinzadeh, Krishnan et al. [10] introduced Gaussian Mixture Models (GMM) in keystroke identification. The authors claimed that keystroke pattern was harder to duplicate as compared to written signature. The reason given was that an imposter had limited number of trials to test on, as most authentication systems would block further access if the verification trial exceeded a certain number of times. A total of 8 subjects were enrolled into their system by typing their full name ten times. The authors also suggested that longer text tended to have lower classification error as it was harder to be reproduced by the others. This is logical as lower number of characters has lower complexity pattern and thus can be easily replicated. Keystroke duration and latency were extracted from the user samples. The Expectation Maximization (EM) algorithm was used to train the GMMs separately using the two extracted keystroke features. Log-likelihood test was performed on the test vector to obtain a probability on how close it was as compared to the user template. The experiment produced superior result of FRR equaled 2.4% and FAR equaled 2.1%. The advantage of their method was on its ability to update the user template upon each successful authentication. However, due to the small number of subjects tested, the result obtained was not decisive.

While conventional timing-based typing characteristics have been widely studied by most of the researchers, [11] investigated the prospect of using typing pressure. They combined the global features of pressure sequences and dynamic time warping with keystroke timing features. The three methods produced separate scores which were then combined using weighted sum rule to obtain the final score. However, they did not indicate how those weights were allocated. In their experiment, 50 samples were provided by each of the 100 users. They were able to obtain an EER of 2.04% by using traditional keystroke dynamics, while the combination with pressure features improved the EER to 1.41%. Although promising result is reported using this approach, pressure sensitive keyboard is not common in real life, making it impractical for large scale deployment. Further more, the percentage of improvement after using pressure sensitive features is fairly insignificant as compared to the increase of cost for the pressure sensitive keyboard.

Guven, Akyokus et al. [12] conducted an experiment on a classifier that resembled neural network like structure for keystroke recognition. This classifier involved timing between successive key strikes (keystroke latency) of 16 users. Usually, weights of neural networks are computed using learning techniques which reduce the variation between an actual output and predicted output. In this study, the weights of the neural network like structure were determined by statistical method which included the mean and standard deviations of the keystroke latencies. Test latency was compared to see if they fall within two times the standard deviation of the reference latency. If all the test latency fulfilled the criteria then the whole string would be considered valid. The experiment recorded poor FRR of 17% and FAR of 26%, which is unfavorable for a practical authentication system. Although the authors attempted to improve the result by using pre-processing methods such as outlier removal, their effort was to no avail.

Hocquet, Ramel et al. [13] studied the possibility of combining three different methods. The first method was using mean and standard deviation of different instances of keystroke latency. The second method deployed the measure of disorder which studied the variation between the time ranks of two signatures. The last method applied time discretization, where each latency was quantized into different levels based on a range of predetermined intervals. Each method formed a matching score which then normalized and combined by using weighted sum rule to obtain the final score. However, no clear indication was given on which method of normalization was used and how the weights were determined. The experiment was carried out on a privately collected database of 38 users consisting of username and passwords with the length between 8 to 30 characters. The performance of the experiment was recorded at an EER of approximately 5%.

In this paper, our two proposed methods Gaussian Probability Density Function (GPD) and Direction Similarity Measure (DSM) are low in complexity and fast in computation. We also propose a fusion approach to merge the scores produced by the said methods which is able to significantly improve the overall result. Besides, our method is able to archive a comparative result compare to those in the literature even with a large number of users involves in the experiments.

III. Methodology

In this section, we first discuss the process of data collection. This is followed by the explanation of keystroke features extracted from the raw keystroke data collected. Next, template generation process will be discussed. We will also explain the two methods for matching the test keystroke data with the reference template.

A. Data Collection

In this research, a program was developed to capture the user keystroke timing using Microsoft Visual Basic 6.0. A total of 100 users are invited to contribute their biometric typing dynamics to the system. During the data collection phase, the users are asked to type his or her favorable username, password and a special fixed line of text ("the brown fox") successively for ten times. This special phrase of text is crucial if we intend to compare different typing patterns among different users.

Two events occur each time the user type a character on the keyboard, to be particular the "key press" and "key release". Each triggered key event will be coupled with a timestamp and these timestamps are kept in plain text files for username, password and special phrase separately. These files are very small in size and do not consume large storage space as they are only normal ASCII files. Figure 1 shows an example of the raw keystroke timestamp of a particular user.

B. Feature Extraction

There are a number of ways to extract and analyze the keystroke feature data from the raw keystroke data collected. Some of the examples are *keystroke latency*, *duration of a key hold*, *pressure of keystroke*, *frequency of word errors*, *typing speed* and *typing difficulty*. Usually, not

```
T,28115607,T,28116920
H,28117180,H,28118417
E,28119039,E,28120012
 ,28121071, ,28122175
B,28122385,B,28123489
R,28125321,R,28126164
O,28127790,O,28129234
W,28130062,W,28131192
N,28132458,N,28132802
 ,28136206, ,28137284
F,28140712,F,28142130
O,28143988,O,28145275
X,28145974,X,28147183
```

Fig.1. Sample keystroke timestamp file.

all of these features will be used for testing purpose. In Figure 2, we illustrate the four types of possible keystroke features. Most of the keystroke features used in the literature are D_1 and D_3. However, we use all of the four features in our experiment.

1) Dwell Time (D_1)

The amount of time on how long a particular key is press down.

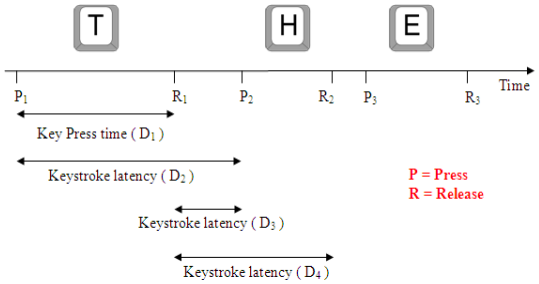

Fig.2. Four types of keystroke features used in our experiment.

$$D_1 = R_1 - P_1 \tag{1}$$

2) Flight Time (D_2)

The time interval between a key press and the successive key press.

$$D_2 = P_2 - P_1 \quad (2)$$

3) Flight Time (D_3)

The time interval between a key release and the successive key press. It may consist of negative value if the successive key is pressed before a key is released.

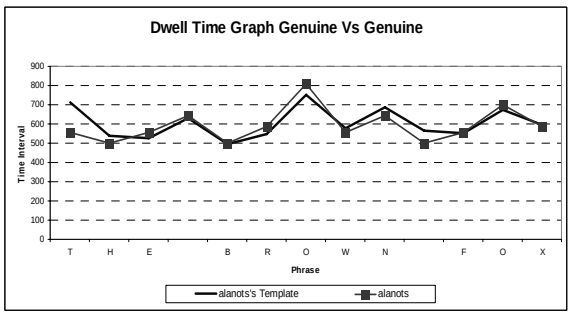

Fig.4. Comparison of dwell time between user reference template and user's own test data.

$$D_3 = P_2 - R_1 \quad (3)$$

4) Flight Time (D_4)

The time interval between a key release and the successive key release.

$$D_4 = R_2 - R_1 \quad (4)$$

C. Template Generation

After obtaining the useful features from the raw keystroke timing data, template generation process will merge and compress all the collected keystroke samples into a compact yet distinctive representation form. These templates comprise of mean and standard deviation of the keystroke feature for each character of username, password and fixed phrase text.

Let a training set of n latencies, $t_1, t_2, ..., t_n$, the mean μ of the set is defined as

$$\mu = \frac{1}{n}\sum_{i=1}^{n} t_i \quad (5)$$

And standard deviation σ is defined as below

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(t_i - \mu)^2}{n}} \quad (6)$$

These generated templates are kept in database and they serves as the reference comparison for future verification uses.

D. Matching

When a new test template is received, a matching function is formulated to calculate the similarity score between the test templates against the stored reference template. We introduce two types of matching function, which are Gaussian Probability Density Function (GPD) and Direction Similarity Measure (DSM). The decision whether to accept a user will base on the score obtained by these functions. A test signature is considered valid if the matching score is above a predefine threshold.

1) Gaussian Probability Density Function (GPD)

The mean and standard deviation in the template and keystroke timing data of a test signature are applied in the function as below.

$$f(t;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}} \quad (7)$$

μ : Mean of a character from reference template
σ : Standard deviation of a character from reference template
t : Keystroke timing data of a character from test signature

Since the function is used to calculate how close the value of reference template and test data template, the output value is a score ranging from zero to one. The closer the score towards the value one, the more similar the test signature will be to the reference template. Hence, we can simplify the function to the form as shown below.

$$Score_{GPD} = e^{-\frac{(t-\mu)^2}{2\sigma^2}} \quad (8)$$

As an example, Figure 3 shows the score of a particular character "Y", the test data's latency is 856 as compared to the reference template which is 819.857. Thus, a similarity score of 0.8045 is obtained. We observe that the closer the latency to the mid of the graph the higher the score achieved, and vise versa. Matching is performed on each and every character in a phrase, which will yield separate individual sub score for each template. The final GPD score

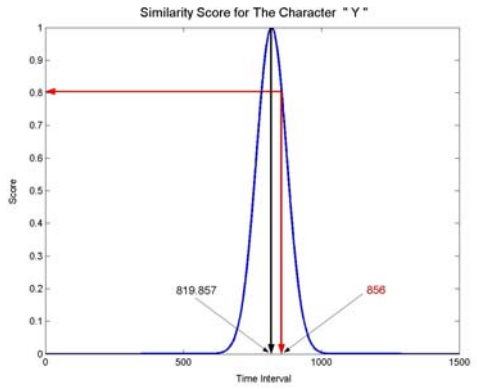

Fig.3. Matching score of a character "Y" obtain by using GPD.

is obtained by calculating the average of all sub scores.

2) Direction Similarity Measure (DSM)

Direction Similarity Measure is a simple yet discriminative approach. The concept behind this method is

to determine the consistency of users' typing behavior. Let ΔD denotes the differentials of D in two consecutive strokes. ΔD represents the strength of the changes and their signs represent the type of the change, i.e. plus sign symbolizes increment, whereas minus sign symbolizes decrement.

For example, Figure 4 shows the comparison of a user's reference template and user's own keystroke input. We observe the change of sign in ΔD. If both signs are the same, we increase the count and vise versa. For example, the difference of 'H-E' in the template is -ve (550-520), while +ve (500-560) for test data. Since the sign is different, it will not be counted. Another example is 'W-N'. The difference of user template is +ve (590-700) while it is +ve (570-640) for the test data, both have the same sign indicating a match. The formula of calculating the Direction Similarity score is defined as follows.

$$Score_{DSM} = \frac{m}{n-1} \tag{9}$$

m: Total matches in a phrase
n : Total characters in a phrase

IV. FUSION

By using fusion technique to combine the matching scores generated by both GPD and DSM, we are able to archive promising result, which is better than using GPD and DSM alone. User reference template will be retrieved from the database to be compared with the claimers' login latency. For each matching process, GPD and DSM are used simultaneously, resulting in two different matching scores. After each matching component produces a partial score, the scores are diverted to the fusion component to produce a final score by using fusion rules. An overview of our fusion approach can be visualized in Figure 5.

Finally a decision (accept/reject) will be made based upon the final score. In the fusion stage, six fusion rules are experimented and each of them is discussed below.

A. Sum rule

Sum rule is one of the simplest and frequently used fixed fusion rules which form an average final score from two given matching scores.

$$Score_{final} = \frac{Score_{GPD} + Score_{DSM}}{2} \tag{10}$$

B. Weighted Sum rule

Instead of assigning a unique weight to each score, a bias weight w is attached to each score for weighted sum rule. It is useful if we wish to emphasize one of the score against the other.

$$Score_{final} = w_1 Score_{GPD} + w_2 Score_{DSM} \tag{11}$$

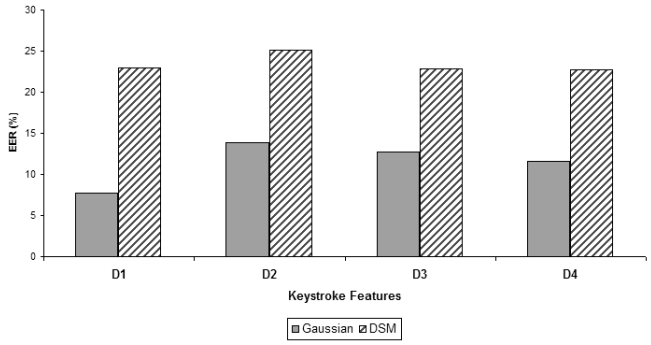

Fig.6. EER of the fusion between GPD and DSM four different keystroke features.

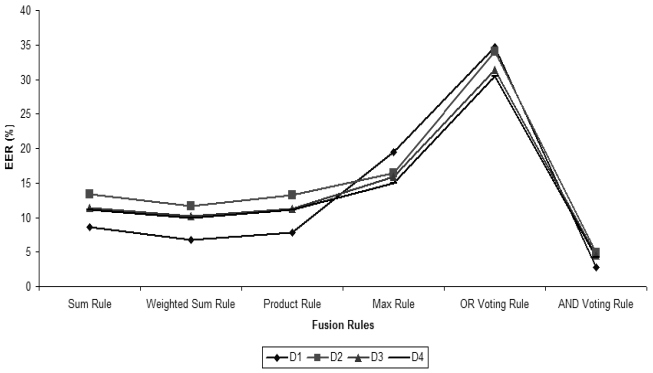

Fig.7. EER of the fusion between GPD and DSM using six fusion rules with four different keystroke features.

C. Product rule

Product rule has a similar concept to sum rule, where by the only difference is that the multiplication operator is used instead of addition.

$$Score_{final} = \frac{Score_{GPD} \square Score_{DSM}}{2} \tag{12}$$

D. Max rule

Instead of computing an overall score value, Max rule does no computation but to only choose an optimal score from the available scores. So, the final score will be the score with the largest value among the given scores.

$$Score_{final} = MAX(Score_{GPD}, Score_{DSM}) \tag{13}$$

E. OR voting rule

OR voting rule does not merge two score values either, instead it merges the decision made upon the individual score. A decision on a score will be accepted if it is higher as compared to a predefine threshold. The final decision will be "accept" as long as any one of partial score fulfills the previous mentioned condition. The only possibility where OR voting rule will produce a reject decision is when both scores are lower than the threshold.

F. AND voting rule

The way AND voting rule works is similar but stricter compared to OR voting rule. It will produce an accept decision only when every scores are above the threshold.

All the other possible combination will result in a negative decision.

V. Experiments and Discussions

In the experiments, testing was done on user template with seven training samples against three testing samples. The testing phase is divided into two parts: (1) Genuine user testing, and (2) Imposter user testing. In the first phase, each user's template is compared with their own testing data (which is the remaining sample data that is not used for template generation). The false rejection rate (FRR) is obtained by taking the ratio of wrongly rejected genuine user and the total number of comparison made. On the other hand, during the imposter user testing, each user template is compared to all the other user testing data. The false acceptance rate (FAR) is obtained by taking the ratio of wrongly accepted imposter and the total number of comparison done. The testing phase is repeated by comparing the test data template score against a threshold in the range 0 to 1 with the interval of 0.01. For each threshold value, we will obtain a value for FAR and FRR. The equal error rate (EER) is obtained where FAR is closest to FRR.

In order to achieve more reliable result, our experiment was performed on randomly selected combination of training and testing data. In each combination, we set ten different orders of combinations with the number of training data is seven and number of testing data is three. The final result is obtained by averaging the EER obtained for each combination of sample data. All the results discussed in the later section are portrayed using this final average EER.

By using only GPD on different keystroke features, we are able to achieve the best EER of 7.72% by using D_1. On the other hand DSM was only able to obtain the best EER of 22.744% using D_4. As we can see in Figure 6, GPD out performs DSM no matter which keystroke feature was used. Since DSM only compares the difference in sign of each coupled character instead of the actual value of the latency, we suspect that this may be the possible reason for the poor performance of DSM. In another aspect, we notice that D_1 is the best choice to be used as keystroke feature. Based on Figure 6 and 7, we can draw a conclusion that by using D_1 as keystroke feature, better performance can be archived for most of the methods tested.

After fusion of GPD and DSM was performed on the six fusion rules, the best result was obtained at EER of 2.791% using the AND voting rule. This result does not only shows a drastic improvement of approximately 20% compare to DSM but also an approximately 5% improvement of GPD. Figure 8 shows the EER comparison between GPD, DSM, and after fusion with the six fusion rules.

We notice that all of the fusion rules at least archive a better result compared to DSM except OR voting rule. The reason behind the big performance gap between OR and AND voting rule might be due to the nature of the two methods GPD and DSM. As we have discussed previously GPD is able to perform better than DSM, in other words the chances of GPD accepting an imposter is low while DSM is higher. Assume a case when GPD rejects an imposter, while DSM has a higher chance to wrongly accept the imposter. If OR voting rule is used, the final decision will be to wrongly accept the imposter. Thus, this results to an overall degradation of performance. On the other hand, if AND voting rule is used, the final decision will be accept only when both GPD and DSM accept a user. Therefore, a stricter condition reduces the chance to wrongly accept an imposter hence increases the overall performance.

Table I summarize the performance comparison between our method and those discussed in the literature. Most of the experiment involves less then 50 subjects, thus the experiment result obtain are not conclusive. The performance obtained by Lv and Wang [11] is the most comparative to ours. They perform their experiment on 100 users and yet still able to obtain a good EER of 1.41%. However, keystroke pressure is used as feature resulting in the need of an additional pressure sensitive keyboard. The additional hardware required might have low acceptance and scalability in real life compared to ours which only uses normal keyboard.

VI. Conclusion and Future Works

As a conclusion, our experiments show that by combining the scores from two methods, Direction Similarity Measure (DSM) with the scores obtained using Gaussian Probability Density Function (GPD), the result can be improved significantly as compared to using them individually. We also show in our experiment that using *dwell time* as keystroke feature leads to better result then all the other keystroke latency. As we can see from the result of this experiment that fusion helps to increase the performance, future work can be directed on fusing more information i.e. combining the information of multiple keystroke features instead of using them separately. Another two issues that could be investigated are the user gradual change of typing patterns as well as some special circumstances when the user is unable to provide their normal typing pattern such as in the case of hand injury. The two problems will affect the performance of a keystroke dynamic recognition system.

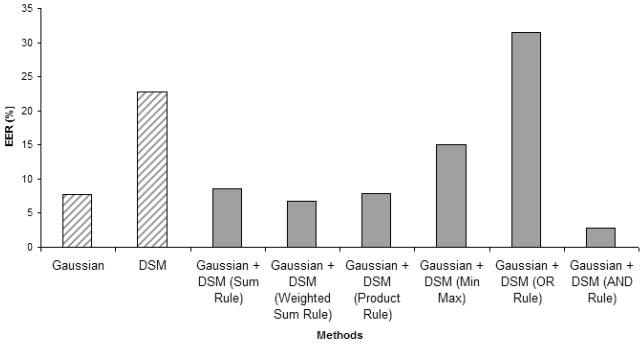

Fig.8. EER comparison between all methods (fusion and non fusion approach).

TABLE I
PERFORMANCE COMPARISON BETWEEN OUR PROPOSED METHODS AND EXISTING METHODS

Papers	Database (subjects)	Features	Result (EER)
Gaines, Lisowski et al. [2]	6	D_3	2%
Umphress and Williams [3]	17	D_3	5.25%
Joyce and Gupta [4]	33	D_3	6.74%
Bleha and Obaidat [5]	10	D_3	8.5%
Cho, Han et al. [6]	21	D_1,D_3	1%
Ord and Furnell [7]	14	D_3	19.95%
Sheng, Phoha et al. [8]	43	D_1,D_3	2.25
Rodrigues, Yared et al. [9]	20	D_1,D_3	3.6%
Hosseinzadeh, Krishnan et al. [10]	8	D_1,D_3	2.15
Lv and Wang [11]	100	D_1,D_3, pressure	1.41%
Guven, Akyokus et al. [12]	16	D_3	21.5%
Hocquet, Ramel et al. [13]	38	D_1,D_3	5%
Our Method	100	D_1,D_2,D_3,D_4	2.791%

REFERENCES

[1] M. S. Obaidat, and B.Sadoun, "Keystroke Dynamics Based Authentication" *in Biometrics: Personal Identification in Networked Society*, Anil Jain et al (Editors), pp. 213-229, Kluwer, MA, 1999.

[2] R. Gaines, W. Lisowski, S. Press, and N. Shapiro, "Authentication by keystroke timing: Some preliminary results", The Rand Report R-256-NSF. Rand Corporation, Santa Monica, CA, 1980.

[3] D. Umphress, and G. Williams, "Identity verification through keyboard Characteristics", *Int. J. Man-Machine Studies*, vol. 23, no. 3, pp. 263-273, Sept. 1985.

[4] R. Joyce and G. Gupta, "Identity authorization based on keystroke latencies", *Commun. ACM*, vol. 33, no. 2, pp. 168–176, Feb, 1990.

[5] S. Bleha and M.S. Obaidat, "Computer User Verification Using the Perceptron", *IEEE Trans. Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 900-902, May/June, 1993.

[6] S. Cho, C. Han, D. Han, and H. Kim, "Web-based keystroke dynamics identity verification using neural network", *Journal of organizational computing and electronic commerce* vol. 10, no.4, pp.295–307, 2000.

[7] T. Ord and S.M. Furnell, "User Authentication for Keypad-Based Devices Using Keystroke Analysis," *Proc. 2nd Int'l Network Conf. (INC 2000)*, Plymouth, UK, 2000, pp. 263–272.

[8] Yong Sheng, V. V. Phoha and S. M. Rovnyak, "A parallel decision tree-based method for user authentication based on keystroke patterns". *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 35, no. 4, pp.826-833, 2005.

[9] Ricardo N. Rodrigues, Glauco F. G. Yared et al., "Biometric Access Control Through Numerical Keyboards Based on Keystroke Dynamics", D. Zhang and A.K. Jain (Editors), *International Conference of Biometrics* (*ICB 2006)*, LNCS 3832, pp.640-646, 2005.

[10] D. Hosseinzadeh, S. Krishnan and A. Khademi, "Keystroke identification based on Gaussian Mixture Models," in *Proc. IEEE Int'l Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2006, vol. 3, pp. 1144-1147.

[11] Lv, H.-R. and W.-Y. Wang, "Biologic Verification Based on Pressure Sensor Keyboards and Classifier Fusion Techniques." *IEEE Transactions on Consumer Electronics*, vol. 52, pp. 1057-1063, August 2006.

[12] Ozlem Guven , Selim Akyokus , Mitat Uysal and Aykut Guven, "Enhanced password authentication through keystroke typing characteristics", *Proceedings of the 25th IASTED International Multi-Conference: artificial intelligence and applications*, Innsbruck, Austria, February 12-14, 2007, p.317-322.

[13] Hocquet, S., J.-Y. Ramel, et al.,"User Classification for Keystroke Dynamics Authentication", *Advances in Biometrics*, Springer Berlin / Heidelberg, pp. 531-539, 2006.