A Study on Keystroke-based Identification

Alberto Palacios Pawlovsky [1] and Noboru Hirabayashi [2]

桐蔭横浜大学工学部電子情報工学科

(2007年3月1日　受理)

Abstract—The increasing power of computers and sensor technology have made possible the implementation of many kinds of biometric systems. In biometric systems, we try to verify or recognize the identity of a user based on his/her physiological or behavioral characteristics. In this work, we show a study about the use of keystroke dynamics for identity recognition. We show some methods for which we have obtained up to 10% of cross-over error rate (CER).

Keywords—Biometrics, identification, keystroke, behavioral, dynamics.

I. Introduction

Biometric systems have shown an incredible proliferation in recent years. Small and powerful computers make now possible to implement many complex algorithms in efficient ways. Sensor technology has also made possible the extraction and processing of neuro-physiological data, and cheap massive storage data equipment make now viable the implementation of real big systems. These systems due to their complexities and expensive maintenance where found only in military installations, highly-secure laboratories of big companies, etc., but now we can find them in USB memories, cellular phones, personal computers, research and development facilities of small companies, and in the control posts to enter many countries. In biometrics systems we have two types of processing, verification and identification. In the verification process the user identifies himself/herself and only the corresponding reference data is used to certify the validity of the user's input data. In identification systems the input data is compared to a set of reference data to determine the corresponding user. We can divide biometric systems in two big categories according to the data they handle. The first category works with human physical data, like a fingerprint, a retina pattern, etc. The second category works with human behavioral traits like gait, signature and typing patterns. Our

(1) Department of Electronics and Information Engineering. Faculty of Engineering, Toin University of Yokohama, 1614, Kurogane-cho Aoba-ku, Yokohama, 225-8502
(2) Now with Business System Planning Corp. of Yokohama, Japan

work deals with the processing of typing patterns. Studies on keystroke latencies and its use in identification has been reported since the late 70's [1], and a lot of work has been done in this area since then. Keystroke-based identification involves many fields that range from the hardware aspects such as the design of special keyboards, software for real time reading of keystroke timing and development of programs to monitor and process the related data. In this work, we only deal with the algorithms to process keystroke data and several ways of determining confidence margins to identify a user. In the following section we describe the identification system of our work. In section III we describe the procedures to handle keystroke data and determine the confidence margins to evaluate it. In section IV we show some ways of handling keystroke data and the cross error rate (CER) we obtained with each of them. In section V we describe briefly an ad hoc approach that divides the users according to the method of determining the confidence margin of their data and the result we can expect with this type of system. In the last section we give some conclusions and topics for future work.

II. Identification System

In our work we used the system shown in Figure 1. We create a pool of reference templates by making each user type the string "toinuniv" and measuring the time each key is pressed and the latencies between keys. The user is required to type this string 30 times for its own template. When one user wants to enter the system he/she is required to type the same string.

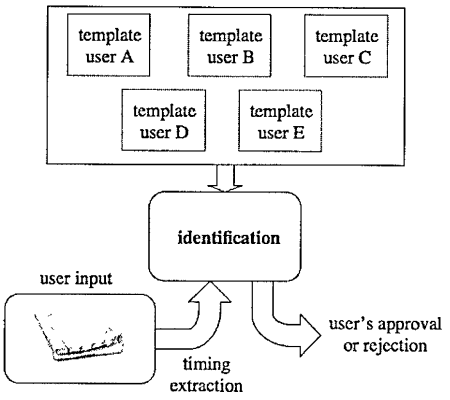

Fig. 1. Block diagram of the identification system of this work.

Our identification system compares the input timing information with those stored in the templates and choose the closest one as the user if it fulfills some requirements.

These requirements indicate how close the input timing information matches that of the template chosen as the most probable one. Usually, this decision is taken based on a threshold that determines the inputs that are accepted and those that are rejected.

If the acceptance threshold is too low unauthorized users will get into the system, and if it is too high even valid users will be rejected and forced to retry entering the system. Usually, this threshold determines the FAR (False Acceptance Rate) and FRR (False Rejection Rate) of the system. Generally, FAR is a monotonic decreasing function, and FRR is a monotonic increasing function on the acceptance threshold (see

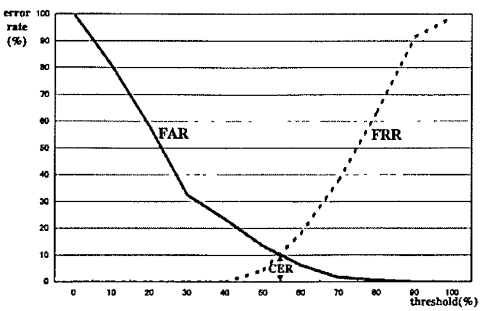

Fig. 2. FAR,FRR and CER.

Figure 2). The point where these two functions take the same value is known as the cross-over error rate (CER). Usually the value of CER indicates the security level of the identification system. Our system is based on the information obtained with the 30 samples of data the user provided to build his/her template. This information is not changed with the data collected when the user enters the system so it is a static identification system. Dynamic systems update the templates with each new data and adapt to the user behavioral changes caused by aging or change of physical characteristics with time. Our system is designed to be used in a personal computer. In a server-client system or systems involving multiple users and where the task of the identification relies on a master computer, exact timing measuring could be a significant problem.

III. Identification Procedure

We implemented a basic input data system in Java that relies on the data provided by the functions this language has to detect keystroke pressing and releasing. Since Java programs use a virtual machine, the data collected is not completely exact due to the overhead created by other tasks of the operating system and by the processing of other statements in the programs themselves. To diminish this influence we limited our computer to run only the data collecting program and no other user program during the template building phase. Under this condition, we collected data from 11 users.

A. Confidence Weights

As indicated above we collected 30 sets of data for each of our 11 users. The timing data between keys pressed is usually called digraphs and extensions to three keys are called trigraphs. We have worked with the timing data between keys and the time the keys are pressed. One sample is shown in Figure 3(a). Usually, the confidence range for an input data is determined around the average of the values used to build the templates. The graph of the average values for some samples is shown in Figure 3(b). In this work we use for comparison the

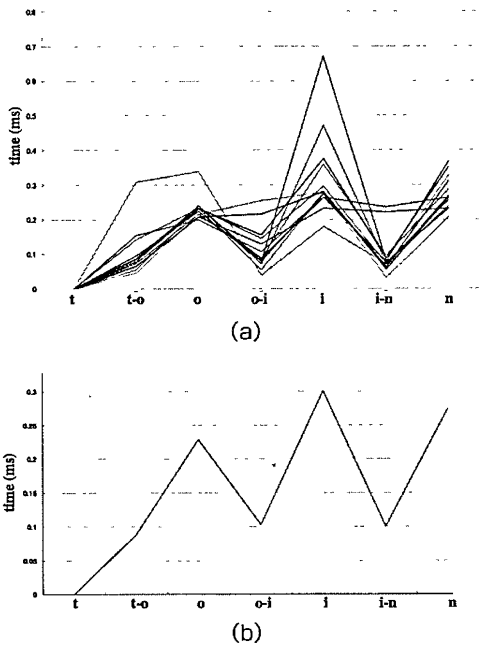

Fig. 3. Template data (a) sample data (b) average value

standard deviation, the average absolute deviation from the mean, and the Euclidean distance. The most basic approach compares the input data with the one of the templates and find whether each value falls within a number of standard deviations from the corresponding average value. If the number of compared data that lie within those boundaries exceeds the acceptance

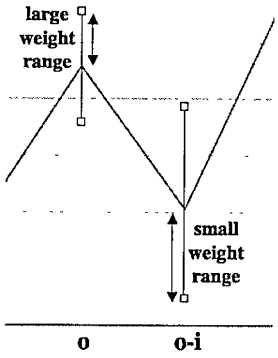

Fig. 4. Range width and confidence weights.

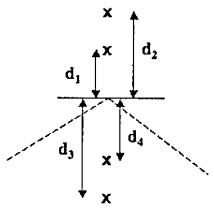

Fig. 5. Euclidean distances to the average value.

threshold of the system, then the input is accepted as valid and corresponding to the user that has the highest achievement in the comparison. Here, when a datum lies within the range allowed by the system, we call it a hit. One way of comparing fitness of an entry to one template is counting the numbers of hits it has. Then, based on it we can identify the user trying to enter the system. The problem with this approach is that when the template data has a high variability the corresponding margins may allow many types of inputs to be considered a hit. To alleviate this problem it has been proposed the use of weights [2]. When the measure use to determine the reference range is far from the average value it will have a small weight and when close a high weight (see Figure 4). In this work we determine the confidence weights based on the average of the Euclidean distances between each datum and the corresponding average value (see Figure 5). One small example is given in Table I. The values are given without units, but they show how we calculate the confidence weights. These confidence weights are expected to be different for different templates (users). These confidence weights are used together with the confidence margins the system use to identify an user.

TABLE I
CALCULUS OF CONFIDENCE WEIGHTS

data	mean $\bar{d}$	$1/\bar{d}$	confidence weight
t	5	0.2	10.26%
o	2	0.5	25.64%
i	4	0.25	12.82%
n	1	1	51.28%
		$\Sigma = 1.95 = 100\%$	$\Sigma = 100\%$

B. Confidence Margins

Many works use the standard deviation to set the margins for each input datum [3], [4]. We also experimented with it to calculate the safety of our system using this measure. Regardless of the measure adopted to set these margins, input data is always compared with the one in the templates to see whether it lies within some given margins. Each time it does counts as a hit (see Figure 6).

In our work we combine the counting of the hits with confidence weights to decide if an input corresponds to a valid user of the

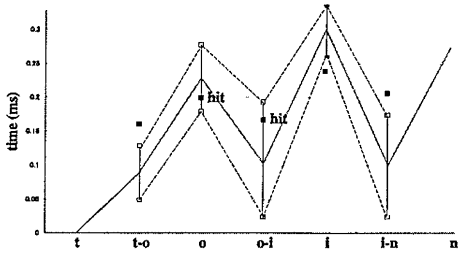

Fig.6. Input example showing only two hits.

system. In Table II we show the confidence weights of the templates of three users of an hypothetical system. In Table III we give as an example the hits obtained with an input to this system and the confidence level obtained summing up the confidence

TABLE II
CONFIDENCE WEIGHTS' EXAMPLE

data	user A	user B	user C
t	30%	25%	15%
o	10%	15%	35%
i	45%	40%	30%
n	15%	20%	20%

weights corresponding to each hit. With these confidence levels, the input would probably correspond to user A (it has the highest confidence level). But its approval depends on the acceptance threshold set in the identification system. If it requires an 80% confidence level, the input will not be accepted as a valid one. However, if it is set to 60% the user would be identified as

TABLE III
CONFIDENCE LEVEL CALULATION EXAMPLE

data	user A	user B	user C
t	hit	-	hit
o	-	hit	-
i	hit	-	hit
n	-	hit	hit
confidence level	75%	35%	65%

either A or C. Notice that user C has more hits than user A but a lower confidence level, so we will probably need an additional rule to choose one of them.

IV. Results with Several Confidence Margins

One way of setting the confidence margin is to use the standard deviation. Using this measure we obtained a CER of approximately 19% for the eleven users of our system (see Figure 7). The values obtained using the average absolute deviation from the mean (also called mean

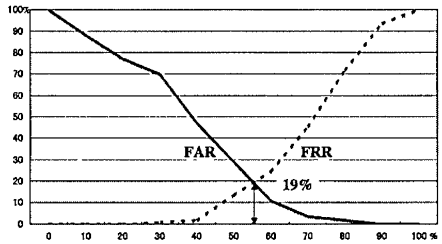

Fig. 7. FAR and FRR when using the standard deviation.

deviation) are shown in Figure 8. Using this measure to set the confidence margins gives a CER of approximately 14%, i.e., 5% better than the CER of the standard deviation.

The values obtained using the mean of the Euclidean distances are shown in

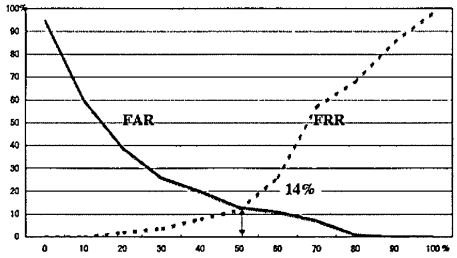

Fig. 8. FAR and FRR when using the average absolute deviation from the mean.

Figure 9. This measure also gave a CER of approximately 14%. However, this value is obtained with an acceptance threshold of nearly 60%, almost 10% higher that the found when using the average absolute deviation from the mean.

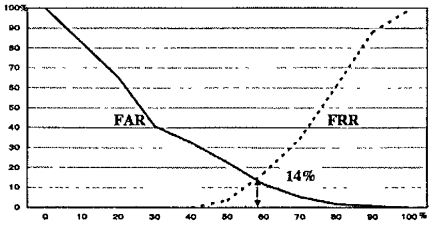

Fig. 9. FAR and FRR when using the Euclidean distance.

Taking this into account we can say that the use of the mean deviation is slightly better. We applied the above three measures in different combinations to see which effect they have combined. In Table IV we show the results we obtained applying each measure and combinations of them to the time the keys are pressed (represented as key in the table) and to the time between keys (represented in the table as k2k). As could be seen from this table not one combination improved the values already shown in Figure 8 or Figure 9.

TABLE IV
CER WITH DIFFERENT MEASURES

key	k2k	CER
(1)	-	34%
(2)	-	16%
(3)	-	33%
-	(1)	27%
-	(2)	21%
-	(3)	34%
(1)	(2)	37%
(1)	(3)	28%
(2)	(1)	39%
(2)	(3)	21%
(3)	(1)	24%
(3)	(2)	27%

(1): standard deviation
(2): mean deviation
(3): mean Euclidean distance

V. An Ad Hoc Approach

Since we could not improve the CER with conventional measures we designed a new one to determine the confidence margins. This is detailed in what follows.

A. 2-region Approach

As shown in Figure 10 we first determine the average value of the data and determine a reference point that is in the middle between the average value and the largest value in the data (see Figure 10(a)). Then, we divide the data in two regions taking as reference this point (Figure 10(b)). Finally, we determine the confidence

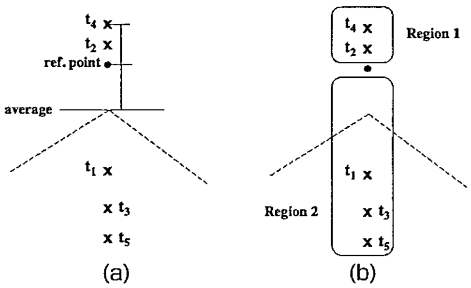

Fig. 10. 2-Region method (a) reference point (b) regions.

range calculating the average of the data in region 1 and region 2 and correspondingly adding and subtracting each one from the average value of all the data. This usually determines two different confidence limits around the average. The CER obtained with this method is shown in Figure 11.

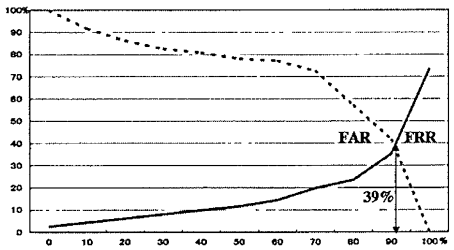

Fig. 11. FAR and FRR when using the 2-region method.

The 2-region method when applied to our system of eleven users did not improve the CER, but showed us that could improve the CER of some particular users (those that had a very low typing speed and data with large dispersion). This hinted us to divide the users in groups according to the method that best fit them. We put three users in a group using the standard deviation, five users in a group using the mean deviation and three users in a group using the 2-region method to determine their confidence range. The result of this ad hoc approach is shown in Figure 12.

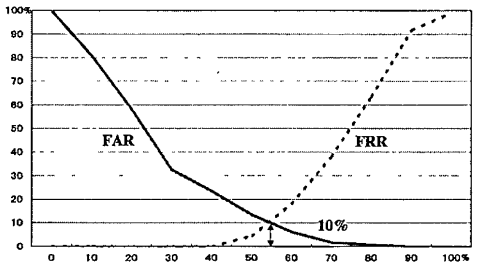

Fig. 12. FAR and FRR when using an ad hoc approach.

All these indicate that to build an identification system with high security (very small CER) we would need to implement it in such a way that the recognition of each user is tailored to the characteristics of his/her data. Well known statistical measures would give good results with some users but it seems that for others we will need to find ad hoc methods such as the one shown here to deal with them properly (with low CERs).

VI. Conclusions

We described in this paper a system to identify an user based on his/her keystroke characteristics. We have shown the details of determining the confidence range with different measures and proposed also a new one. As was shown above the best of our approaches gives a CER of 10%. This seems to be not good enough, but we have to take into account that we have worked with only one string. We can improve the security level if we require each user to type his/her own password, but this will change our system into one of only verification. Other works suggest that increasing the number of strings the user has to type also increases the performance. This and other possible approaches will be matter of future research.

Acknowledgments

We thank to the members of our laboratory that gently provided the data for this work.

References

[1] R. Spillane, "Keyboard Apparatus for Personal Identification," IBM Technical Disclosure Bulletin, Vol. 17, No. 3346, 1975.

[2] Darren Clifford D'Souza, "Typing Dynamics Biometric Authentication," Bachelor thesis, University of Queensland, Australia, October 2002.

[3] F. Bergadano, D. Gunetti, and C. Picardi, "User Authentication Through Keystrokes Dynamics," ACM Transactions on Information and System Security, Vol. 5, No.4, pp. 367-397, November 2002.

[4] A. Peacock, X. Ke, and M. Wilkerson, "Typing Patterns: A Key to User Identification," IEEE Security and Privacy, September/October 2004, pp.40-47.