Improving versatility in keystroke dynamic systems

Enrique Calot, Juan Manuel Rodriguez, and Jorge Salvador Ierache[*]

Laboratorio de Sistemas de Información Avanzados,
Facultad de Ingeniería, Universidad de Buenos Aires,
Buenos Aires, Argentina
{ecalot,jmrodri}@fi.uba.ar,jierache@yahoo.com.ar

Abstract. Keystroke dynamics is a biometric technique to identify users based on analyzing habitual rhythm patterns in the way they type. In order to implement this technique different algorithms to differentiate an impostor from an authorized user were suggested. One of the most precise method is the Mahalanobis distance which requires to calculate the covariance matrix with all that this implies: time processing and track each individual keystroke event. The hypothesis of this research was to find an algorithm as good as Mahalanobis which does not require track every single keystroke event and improve, where possible, the processing time. To make an experimental comparison between Mahalanobis distance and euclidean normalized, a distance which only requires calculate the variance, an already studied dataset was used. The results were that use normalized euclidean distance is almost as good as Mahalanobis distance even in some cases could work better.

Keywords: Keystroke Dynamics, Web based authentication, Mahalanobis distance, Biometrics, typing biometrics

1 Introduction

The variables that help make a handwritten signature a unique human identifier also provide a unique digital signature in the form of a stream of latency periods between keystrokes. The handwritten signature has a parallel on the keyboard. The same neurophysiological factors that make a written signature unique are also exhibited in a user's typing pattern[1].

Password typing is the most widely used identity verification method in World Wide Web based electronic commerce. Due to its simplicity, however, it is vulnerable to impostor attacks. Keystroke dynamics and password checking can be combined to result in a more secure verification system[2].

This authentication is fragile when there is a careless user and/or a weak password. Biometric characteristics are unique to each person and have advantages as they could not be stolen, lost, or forgotten[3, 4].

[*] This paper was done with Cloodie R&D Support

The biometric technology employed in this paper is the typing biometrics, also known as keystroke dynamics. Typing biometrics is a process that analyzes the way a user types at a terminal by monitoring the keyboard inputs in attempt to identify users based on their habitual typing rhythm patterns[5, 4].

Even though WWW keystroke dynamic systems may run locally on the web browser, due to security measures it should be ran on the webserver layer. This paper discusses an approach to reduce data transmission size.

Using a know dataset[6] we designed an experiment to compare three methods to compute the keystroke dynamics of users and compare them with impostors.

Our hypothesis is that one of the most used and precise method –the Mahalanobis distance– is as successful as the second method –normalized euclidean distance–. Ignoring the success rates, there are some advantages that the normalized euclidean distance has over the Mahalanobis distance, so if the hypothesis is confirmed using this method should prove to be a more useful way to calculate keystroke dynamics.

Some advantages of the normalized euclidean distance ar the lesser transferred information, processing time and the bigger versatility when changing passwords.

2 Current implementations

There are different methods to compare keystrokes, all based on measuring the distances between two strokes, a negative result is found when both differ more than a threshold. One of the best methods is the Mahalanobis distance[2, 6].

Three methods are shown below, each method is a generalization of the previous one.

2.1 Euclidean distance

The time a user press a key or the time between one key and the other may result in a vector of events ($\boldsymbol{\Gamma}$). Each event represent a key hold time or the elapsed time between two keys. Since in the training phase an event may occur several times, the vector is a list of the expected values of every event time.

Calculating the euclidean distance between two vectors works as a comparison algorithm, with relatively high success rates[6].

$$d(\boldsymbol{\Gamma_1}, \boldsymbol{\Gamma_2})^2 = \|\boldsymbol{\Gamma_1} - \boldsymbol{\Gamma_2}\|^2 = \sum_{i=1}^{N} (\Gamma_{1,i} - \Gamma_{2,i})^2 \tag{1}$$

Where $\boldsymbol{\Gamma_1}$ is the vector of training event times and $\boldsymbol{\Gamma_2}$ is the vector of testing event times.

To optimize calculation timings the squared norm is actually used.

2.2 Normalized euclidean distance

A disadvantage of the former method is that important information is ignored. The variance of each event time should be taken into consideration, and that is exactly what the normalized euclidean distance does: adding the variance (s_i^2) of each event time.

Using the inverted variance of the training set ($\boldsymbol{\Gamma_1}$) as a weight factor, the normalized euclidean distance is defined as

$$d(\boldsymbol{\Gamma_1}, \boldsymbol{\Gamma_2})^2 = \sum_{i=1}^{N} \frac{(\Gamma_{1,i} - \Gamma_{2,i})^2}{s_i^2} \tag{2}$$

where s_i^2 is the variance of each element of $\boldsymbol{\Gamma_1}$.

2.3 Mahalanobis distance

Again, the former method is skipping information, this time is the covariance between events.

Mahalanobis distance is defined as

$$d(\boldsymbol{\Gamma_1}, \boldsymbol{\Gamma_2})^2 = (\boldsymbol{\Gamma_1} - \boldsymbol{\Gamma_2})^T S^{-1} (\boldsymbol{\Gamma_1} - \boldsymbol{\Gamma_2}) \tag{3}$$

Where S^{-1} is the inverted covariance matrix corresponding to all events in the training set $\boldsymbol{\Gamma_1}$[7].

3 Problems of Mahalanobis distance in keystroke dynamics

Translating each method to a kernel matrix it turns out that in equation 3 the matrix S is the identity in the euclidean distance, a diagonal with the variances in the normalized euclidean distance and the covariance matrix in the Mahalanobis distance.

3.1 Distance kernel matrix size

To generate the covariance matrix for the Mahalanobis distance all key-press events and their respective timings should be transmitted to the server while training –or at least the covariance matrix and the expected event timings–. But to generate the diagonal matrix for the normalized euclidean method is it possible to send only three integer numbers per event or two floats.

Using the property $Var[X] = E[X^2] - E[X]^2$ it is possible to generate the variance using only the sum of squares $SS = \sum_{i=0}^{n} \Gamma_{1,i}^2$, the sum $S = \sum_{i=0}^{n} \Gamma_{1,i}$ and the total n since $E[X^2] = \frac{SS}{n}$ and $E[X] = \frac{S}{n}$. All three numbers are natural and may be combined in an $\mathbb{N}^3$ vector which supports commutative addition properties. This method allows parallelized variance calculus[9].

Table 1. Different parameters to be sent to the server

Distance Method	Variables
Euclidean	$(S, n) \in \mathbb{N}^{2\times n}$ or $\mathit{\Gamma} = E[X] \in \mathbb{R}^n$
Normalized euclidean	$(S, SS, n) \in \mathbb{N}^{3\times n}$
Mahalanobis	$\mathit{\Gamma} = E[X] \in \mathbb{R}^n$ and $Cov[X] \in \mathbb{R}^{n\times n}$

There are several ways to send the data depending on the algorithm to be used. Table 1 compares them.

For example, when 20 events are used, the covariance matrix has $20\times 20 = 400$ values and the $\mathit{\Gamma} = E[X]$ vector has 20 values. Normally a $\mathbb{R}^{n\times n}$ matrix can be encoded with n^2 numbers, but as $Cov[a, b] = Cov[b, a]$, covariance matrix is symmetric and therefore it can be encoded with $\frac{n(n+1)}{2}$ numbers. Assuming a real number and an integer has the same size, the transmission would be of $\frac{20\times 21}{2} + 20 = 230$ numbers while the normalized euclidean only transmits $3 \times 20 = 60$ numbers. Generalizing, the data transmission of Mahalanobis distance is $\frac{n(n+1)}{2} + n$ reals, that is $\mathcal{O}(n^2)$, normalized euclidean is $3n$ integers, that is $\mathcal{O}(n)$ and euclidean is $2n$ integers or n reals, that is also $\mathcal{O}(n)$.

3.2 One password algorithm

Another problem is that training is done with only one password. A new password should require the user to re-train all the covariance matrix with Mahalanobis.

Normalized euclidean distance may reuse the variances of the common keys between two different passwords while Mahalanobis distance may not.

3.3 Backspace eliminating digraphs

When the user trains it is possible that mistakes a character and use backspace to correct it, in this case one event will be missing. For example the word "train" has 5 characters so the events will be `t.hold`, `t.up-r.down`, `r.hold`, etc. The problem occurs when "te`[backspace]`rain" is typed, the event `t.up-r.down` was not recorded because there were two keys in the middle "e" and `[backspace]`.

Having a variable number of events per key is a problem to calculate the covariance matrix, but allows to process backspaces in passwords (sacrificing the success rate due to lesser information available) and free text.

Table 2 shows an example of Mahalanobis method with three pairs of events and normalized euclidean with three and two times per event respectively.

3.4 Processing times

Calculating the covariance matrix and inverting it should take a considerable amount of time for the Mahalanobis method, the time here is expected to be

Table 2. Example of how timing counts are dependent on the event in Mahalanobis distance

Method	Key	Times	Matrix S	Inverse S^{-1}
Mahalanobis	`A.hold` `A.up-B.down`	$\left\{\begin{matrix} 90 \\ 161 \end{matrix}\right\}, \left\{\begin{matrix} 99 \\ 171 \end{matrix}\right\}, \left\{\begin{matrix} 97 \\ 174 \end{matrix}\right\}$	$\begin{bmatrix} \frac{67}{3} & \frac{175}{6} \\ \frac{175}{6} & \frac{139}{3} \end{bmatrix}$	$\begin{bmatrix} \frac{556}{2209} & -\frac{350}{2209} \\ -\frac{350}{2209} & \frac{268}{2209} \end{bmatrix}$

Normalized euclidean	`A.hold`
	`A.up-B.down`
		{90}, {99}, {97}
		{161}, {171}
			$\begin{bmatrix} \frac{67}{3} & 0 \\ 0 & 50 \end{bmatrix}$	$\begin{bmatrix} \frac{3}{67} & 0 \\ 0 & \frac{1}{50} \end{bmatrix}$

$\mathcal{O}(n^2)$. Normalized euclidean should also take time to compute the variances, but this procedure is $\mathcal{O}(n)$. Inverting the matrix lacks of relevant costs due to the properties of the diagonal matrices. Euclidean distance should be the fastest algorithm because of its simplicity.

It is important to remark that due to parallelized calculation of the variance, part of the calculating time in training mode for the normalized euclidean distance may be done while reading the keyboard by the user machine.

The experiment also intends to measure algorithms processing time.

4 Experimental comparison

We use an already studied dataset for two main reasons, one is because it was collected in a controlled laboratory environment, the second reason is because 14 detection algorithms were tested using this dataset[6] and that give us a big framework to start our research. The data was collected from 50 different users along 8 days or sessions –totalizing 400 cases per user–, in each session the users typed always the same string: "`.tie5Roanl`" which represents a reasonable secure password. When any error in the sequence was detected, the subject had been prompted to retype the password. To make this a laptop was set up with an external keyboard to collect data and a Windows application was developed to prompt the subjects to type the password. The application displays it in a screen with a text-entry field. In order to advance to the next screen, the subject must type the 10 characters of the password correctly in sequence and then press Enter. The data set contains the hold time of each key, the time between two consecutive keys were pressed and the time since one key was released and the next was pressed. One of the three values depends linearly of the other two. Due to preconditions of covariance one value was dropped away leaving two values per key.

From the 400 cases per user, the first 200 cases were used to train the detection algorithm and the second 200 cases were used to validate it, also the first 5 cases of all the other users were taken to generate an impostor dataset in order to validate negative cases. This data set and schema was taken from Killourhy and Maxion[6].

We performed 19 tests, the first using two events (the first two values of the Γ vector) and increasing the number of events until the last one, using all twenty.

We expected to have a best success rate in the last test because it counts with more information. We ran the three mentioned algorithms in each test.

Finally we calculated the area under the receiver operating characteristics (ROC) curve –a performance measure for machine learning algorithms commonly used in systems that learns by being shown labeled examples[8]–. This method, known as AUC, was chosen because it is a classifier performance evaluator independent of the decision threshold chosen on the keystroke distances.

5 Results

With the one key test case we obtained in one sample user $\boldsymbol{\Gamma} = [98.98, 166.905]$, where first value corresponds to the expected key-hold time and the second to the expected elapsed time until the next key was pressed. Both times are expressed in milliseconds.

$$S^{-1}_{Mahalanobis} = Cov[\boldsymbol{\Gamma}]^{-1} = \begin{bmatrix} 341.29 & 282.19 \\ 282.19 & 5464.9 \end{bmatrix}^{-1} = \begin{bmatrix} 0.0031 & -0.00016 \\ -0.00016 & 0.0002 \end{bmatrix}$$

$$S^{-1}_{normalizedEuclidean} = \begin{bmatrix} 341.29 & 0 \\ 0 & 5464.9 \end{bmatrix}^{-1} = \begin{bmatrix} 0.0029 & 0 \\ 0 & 0.00018 \end{bmatrix}$$

$$S^{-1}_{euclidean} = S_{euclidean} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Note that $S_{Mahalanobis}$ and $S_{normalizedEuclidean}$ have the same diagonal values but this is not the case with their inverses.

Table 3. Experimental results

N	Method	Total Errors	ROC	Zero-miss	False-Alarm	Time
2	Mahalanobis	0.01887	80.43%	7461 of 12750	769 of 10200	1.356s
2	Normalized euclidean	0.01899	80.17%	7451 of 12750	788 of 10200	1.300s
2	Euclidean	0.02240	70.61%	9030 of 12750	649 of 10200	0.872s
20	Mahalanobis	0.00970	94.60%	5576 of 12750	464 of 10200	1.896s
20	Normalized euclidean	0.00919	94.84%	5581 of 12750	428 of 10200	1.764s
20	Euclidean	0.01440	88.27%	6853 of 12750	844 of 10200	1.704s

Table 3 shows the results of the 3 methods with 2 and 20 timing events respectively. Each method shows the area under ROC curve in percentage among with zero-miss and false-alarm rates. It is also shown the total processing time of training, testing all the 12750 positive and 10200 negative sets and calculating the results.

In the last test –with 20 events–, normalized euclidean distance method performed even better than Mahalanobis.

As expected, our hypothesis that in the test with 20 timing events is better than the test with 2 was confirmed and that Mahalanobis and normalized euclidean distance are both superior than euclidean distance was confirmed too. Processing is, as expected, bigger for Mahalanobis and decreasing for normalized euclidean and finally, the fastest method, euclidean distance.

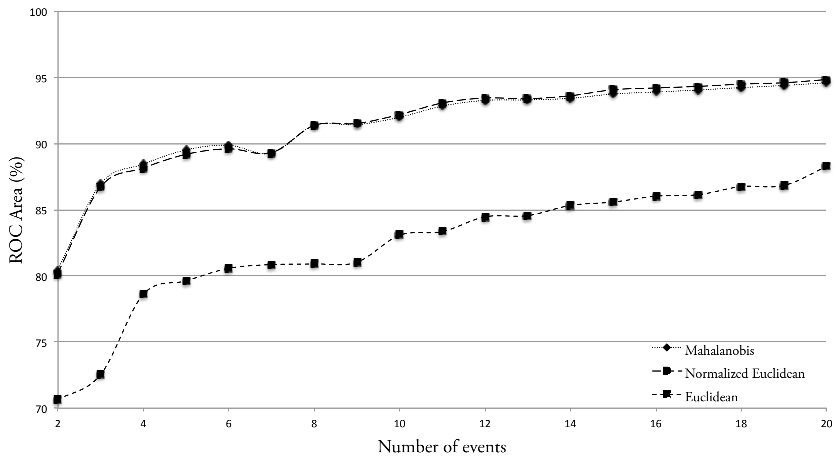

Fig. 1. Distance methods compared in success versus amount of information

In Figure 1 it is possible to appreciate how similar are the normalized euclidean and Mahalanobis distances compared to the euclidean.

6 Conclusions

Normalized euclidean distance and Mahalanobis distance are almost the same in all ran tests. In the case of 20 events the results varied 0.24%. Normalized euclidean was faster than Mahalanobis distance for 132ms but slower than euclidean for only 60ms. Versatility in normalized euclidean is also an advantage, passwords may be changed and the already-trained keys be kept in the new training. Those results lead to the conclusion that normalized euclidean distance is strong enough to be used and its advantages in data sizes and versatility are considerably important to be chosen against Mahalanobis distance and its success rate suggests that it should be employed against euclidean distance.

6.1 Future lines of research

We are exploring the way users may vary keystroke dynamics over the time. Using variance parallelization principle[9] there is a way to "forget" the training, making it autoadaptive with this time-wise learning technique. We are also exploring new fields on keystroke dynamics that include user emotional state detection.

Acknowledgments. This paper acknowledges support from Cloodie R&D.

References

1. Joyce, R.; Gupta, G.: Identity authentication based on keystroke latencies. Commun. ACM 33, 2 (February 1990), 168-176. `http://doi.acm.org/10.1145/75577.75582` (1990)
2. Cho, S.; Han, C.; Han., D. H.; Kim, H. I.: Web based Keystroke Dynamics Identity Verification using Neural Network. Journal of Organizational Computing and Electronic Commerce, Vol. 10, No. 4, pp. 295–307 (2000)
3. Polemi, D.: Biometric Techniques: Review and Evaluation of Biometric Techniques for Identification and Authentication, Including an Appraisal of the Areas Where They are Most Applicable. Institute of Communication and Computer Systems, National Technical University of Athens, Athens, Greece. Retrieved on 2013-07-01: ftp://ftp.cordis.lu/pub/infosec/docs/biomet.doc, EU Commission Final Rep. (1997)
4. Araujo, L. C. F.; Sucupira, L. H. R.; Lizarraga, M. G.; Ling, L. L.; Yabu-Uti, J. B. T.: User authentication through typing biometrics features. Signal Processing, IEEE Transactions on, vol. 53, no. 2, pp.851–855 (2005)
5. Monrose, F.; Rubin, A. D.: Keystroke dynamics as a biometric for authentication. Future Gen. Comput. Syst., vol. 16, no. 4, pp. 351-359 (2000)
6. Killourhy, K. S.; Maxion, R. A.: Comparing Anomaly-Detection Algorithms for Keystroke Dynamics. In International Conference on Dependable Systems & Networks (DSN-09), pp. 125–134, Estoril, Lisbon, Portugal, 29 June to 02 July 2009. IEEE Computer Society Press, Los Alamitos, California (2009)
7. Mahalanobis, P. C.: On the generalised distance in statistics. Proceedings of the National Institute of Sciences of India 2 (1): 49-55. Retrieved on 2013-07-01: `http://www.new.dli.ernet.in/rawdataupload/upload/insa/INSA_1/20006193_49.pdf` (1936)
8. Bradley, A. P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition, Volume 30, Issue 7, July 1997, Pages 1145–1159, ISSN 0031-3203, `http://dx.doi.org/10.1016/S0031-3203(96)00142-2`. (1997)
9. Chan, T. F.; Golub, G. H.; LeVeque, R. J.: Updating Formulae and a Pairwise Algorithm for Computing Sample Variances. Technical Report STAN-CS-79-773, Department of Computer Science, Stanford University. Retrieved on 2013-07-01: `ftp://reports.stanford.edu/pub/cstr/reports/cs/tr/79/773/CS-TR-79-773.pdf` (1979)