Keystroke Dynamics in a General Setting

Rajkumar Janakiraman and Terence Sim

School of Computing, National University of Singapore, Singapore 117543
{rajkumar,tsim}@comp.nus.edu.sg

Abstract. It is well known that Keystroke Dynamics can be used as a biometric to authenticate users. But most work to date use fixed strings, such as userid or password. In this paper, we study the feasibility of using Keystroke Dynamics as a biometric in a more general setting, where users go about their normal daily activities of emailing, web surfing, and so on. We design two classifiers that appropriate for one-time and continuous authentication. We also propose a new Goodness Measure to compute the quality of a word used for Keystroke Dynamics. From our experiments we find that, surprisingly, non-English words are better suited for identification than English words.

Keywords: Keystroke dynamics, biometrics.

1 Introduction

Keystroke Dynamics is increasingly being used as a biometric for user authentication, no doubt because keyboards are common input devices, being readily found on computers, telephones, ATM machines, etc. By Keystroke Dynamics we mean the temporal typing pattern (the way you type), rather than the content typed (what you type). Most of the research into Keystroke Dynamics, however, is done on fixed-text input, otherwise called *password hardening* [3,4,6,10], rather than on free text. Typically, keystroke authentication is performed during user-login on a pre-determined string, such as the userid or password. This seems to us to be somewhat limiting, considering that most people continue to use the keyboard well beyond user-login. It would certainly be more useful if Keystroke Dynamics can handle free text as well as fixed text.

In our literature search, we note that S.J. Shepherd [1] was perhaps the first to explore using Keystroke Dynamics for continuous authentication, using the rate of typing. The system authenticated the user based only on the mean and standard deviation of the Held Times and the Interkey Times, irrespective of the key being pressed. Although it worked for a user population of four, the accuracy of the system is likely decrease as the number of users increase. There is no guarantee that these features are sufficiently discriminative. Indeed, our experiments conducted with a larger pool of 22 users confirm this.

Recent works of Villani et al., Rao et al., and Leggett et al. [7,8,9], conducted studies on keystroke verification on fixed text as well as free text. The users were asked to type a pre-determined text of a few hundered keystrokes (much longer

S.-W. Lee and S.Z. Li (Eds.): ICB 2007, LNCS 4642, pp. 584–593, 2007.

than the usual userid and password), and a text of a few hundred keystrokes of their own choice in the keystroke capture application. This data is then used for training and testing of their verification systems. The general conclusion from their studies is that Keystroke Dynamics works better on fixed text than on free text. We remark that these researchers all used Held Times and Interkey Times (of up to three consecutive keystrokes) as features, and did not consider the actual words being typed. We believe this is the cause of their poor performance. Our work in this paper suggests that Held Times and Interkey Times do indeed depend on the words typed. That is, the timings for 'THE' is different for 'FOR'. By using word-specific Held and Interkey Times, we are able to achieve greater accuracy. In other words, we are using fixed strings within free text for the purpose of discrimination. We show that many fixed strings qualify as good candidates, and this allows us to verify the user as soon as any of these strings are typed.

Can a sample of keystroke data identify a user without any constraints on language or application? In other words, we wish to identify a person without any constraint on what he or she types. The person is not required to input a pre-determined text. Can Keystroke Dynamics still be used in such a general setting? In this paper, we attempt to answer this question. The answer will help in the design of *continuous authentication* systems [5], in which the system continuously checks for the presence of the authorized user after initial login. In such a scenario, it is impractical to demand the user to repeatedly type her userid or any other pre-determined text. Instead, the system has to utilize the typing patterns present in free text for authentication.

Perhaps the work closest to ours is that of Gunetti and Picardi [12], in which clever features were devised (along with suitable distance metrics) for free-text authentication. More precisely, Gunetti and Picardi avoided using the usual digraph and trigraph latencies directly as features. Instead, they used the latencies only to determine the relative ordering of different digraphs, and devised a distance metric to measure the difference between two orderings of digraphs, without regard to the absolute timings. The authors reported a False Accept Rate (FAR) of 0.0456% at a False Reject Rate (FRR)[1] of 4.0%, which, although worse than their fixed-text system, is state of the art for free-text systems.

We begin by analyzing the keystrokes of users as they go about their normal daily activities of emailing, web surfing, etc. We then look for patterns that can be used as a biometric. Such a pattern has to be discriminative, and at the same time common (universal) across all users (because the pattern cannot be used on people who do not type it). Also, for practical purposes, we should not have to wait too long for such a pattern to appear. The pattern should be readily available. We discover that, indeed, discriminative, universal and available patterns do exist even when the typing is unconstrained. Moreover, non-English words are better suited for this task. As far as we can tell, we are the first to investigate the problem of Keystroke Dynamics in a general setting. Our paper makes the following contributions:

[1] In the keystroke dynamics literature, FRR and FAR are also known as the False Alarm Rate and the Imposter Pass Rate, respectively.

1. We propose a new Goodness Measure to assess a keystroke pattern based on its discriminability, universality, and availability.
2. We show that Keystroke Dynamics can be used as a biometric even in a general setting.
3. We show that, surprisingly, some non-English words have a higher Goodness Measure than English words.
4. We propose two classifiers that are suitable for one-time and continuous keystroke authentication.

2 Basic Concepts

In this section we explain the basic terminology used in this paper. Fig. 1 shows a typical keystroke stream, collected from a user. Each arrow indicates a keyevent - the down facing arrow indicates a key being depressed, the upward facing arrow indicates the key being released. A pair of keyevents, a press and a release of the same key, form a keystroke.

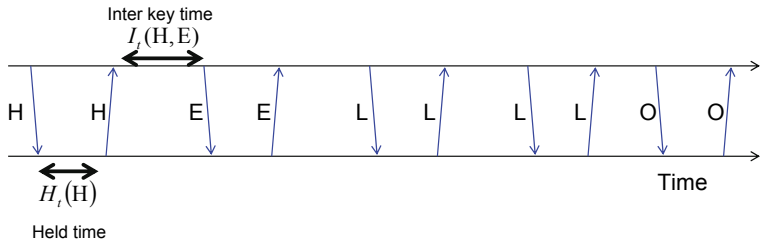

Fig. 1. Typical keystroke data. Each upward and downward pointing arrow indicates a keyevent.

2.1 Definitions

Held Time (H_t)**.**We define Held Time as the time (in milliseconds) between a key press and a key release of the same key. Fig. 1 shows how the Held Time of the key 'H' is determined. Note that Held Time is strictly greater than zero.

Interkey Time (I_t)**.** This is defined as the time in milliseconds between two consecutive keystrokes. In Fig. 1, I_t(H,E) is the time between the key release of 'H' and key press of 'E'. Interkey Times can be negative, i.e. the second key is depressed before the first key is released.

Sequence. We define Sequence as a list of consecutive keystrokes. For example 'HELLO' in the Fig. 1 is a Sequence. A Sequence can be of any length, the minimum being two. In this example, the Sequence is a valid English word, but this need not be the case. Thus, 'HEL', 'LLO' are also valid Sequences from the same keystroke stream in Fig. 1.

Feature Vector (F_t)**.** This is a vector of the Held Times followed by the Interkey Times of a Sequence. For the Sequence 'THE', its feature vector is:

$$F_t(\text{THE}) = \left[H_t(\text{T}) \;\; H_t(\text{H}) \;\; H_t(\text{E}) \;\; I_t(\text{T,H}) \;\; I_t(\text{H,E}) \right]^\top \tag{1}$$

For a Sequence of length n, the length of the feature vector will be $2n - 1$.

2.2 Histogram (Hist_{seq})

From the samples of the Sequence appearing in the keystroke data, we estimate the probability density function (pdf) for each element in the feature vector F_t. This information is stored as a normalized histogram for the Sequence, which in turn will be used for classification.

We choose to represent the pdf as a histogram rather than as the parameters of a multidimensional Gaussian pdf because we observe that the data are rarely normally distributed. It is well known that a histogram with a fixed bin size is able to represent any pdf more accurately than a single Gaussian distribution.

Given two pdfs h_i, h_j, how similar are they? This may be measured using the Bhattacharyya distance or the Kullback-Leibler divergence [11]. We prefer the Bhattacharyya distance [11] because of its symmetry:

$$Dist_B(h_i, h_j) = \int \sqrt{h_i(\mathbf{x}) h_j(\mathbf{x})} d\mathbf{x} \tag{2}$$

This distance is always between 0 and 1, with 1 meaning that the two pdfs overlap perfectly, and 0 meaning that the pdfs do not overlap at all. Since our pdfs are discretized as histograms, the actual computation of Equation (2) is performed by first multiplying the corresponding bins of the two histograms, taking the positive square root of the products, and then summing over all the bins. We will use the Bhattacharyya distance in Classifier B (see Section 2.3).

2.3 Classifier

Classifier A is designed to identify a person from a single instance of a Sequence appearing in the keystroke data. The identity of the person is given by

$$\arg \max_{person} P(\text{person} \mid \text{seq}) \tag{3}$$

where,

$$P(\text{person} \mid \text{seq}) = \prod_{f \in F_t} P(\text{Hist}_{seq}^{person} \mid f) \tag{4}$$

Here we are making the Naïve Bayes assumption, i.e. the elements of the Feature Vector F_t are statistically independent. Classifier A is useful for applications where authentication is required immediately without any delay; for example, in a login module that prompts the user to type a system-generated string (such a string is different each time, to guard against replay attacks).

Classifier B is designed to identify a person from multiple instances of the same Sequence appearing in the keystroke data. Here we first build a histogram ($Hist_{in}$) from the input keystroke stream and then compare it with the learned histogram ($Hist_{seq}$) using the Bhattacharyya distance. The identity of the person is given by

$$\arg \max_{person} Dist_B(\text{Hist}_{seq}, \text{Hist}_{in}) \tag{5}$$

Again, we make the Naïve Bayes assumption. Classifier B is useful for applications which can afford to wait and collect enough keystrokes (thereby accumulating more evidence) before authentication.

3 Experiments

All our experiments are based on keystroke data collected from 22 users over a period of two weeks. The users are staff or students from our department, with different typing abilities. Some are trained typists, in that they had undergone typing classes and could type without looking at the keyboard. Others are untrained typists, but are still familiar with the keyboard as they have used it for many years. The users are of Chinese, Indian or European origin, and all are fluent in English.

Unlike most other studies, the data we collected were not controlled by any means. Keystrokes were logged as users went about their daily work of using email, surfing the web, creating documents, and so on. The collected data from individual users ranged from 30,000 keyevents to 2 million keyevents. In total 9.5 million keyevents were recorded. The PCs used belonged to each individual user, that is, they were not shared machines nor public access computers. Most PCs had keyboards with the Dell^{TM} US layout, although a few users used their own non-Dell laptops. Each user used the same keyboard throughout the data collection, and thus the problem of different keyboards affecting their typing speed did not arise.

To collect keystrokes, we wrote a data collector program in Visual C. This was basically a System Wide Keyboard and Mouse hook, which collects keyboard and mouse events regardless of the application. The data collector was installed in the user's machine running Microsoft Windows XP^{TM} . When activated the program collects all the keyevents and the mouseevents along with the timestamp and name of the application receiving the event. To protect the privacy of the users, each userid and machine id were hashed to a unique number. Also, a shortcut key was provided so that the user can switch it off while typing sensitive information such as passwords or pin numbers.

Table 1. The ten most frequently used English words, in descending order of frequency

THE, OF, AND, A, TO, IN, IS, YOU, THAT, IT

3.1 Results I - For Sequences That Are English Words

Classifier A and Classifier B were run on selected Sequences from the keystroke data that are English words. The words are selected from a Corpus [2] of most frequently appearing English words, see Table 1. The user data was split into 10 bins and 10-fold-cross-validation was conducted with both classifiers. The presented results are the mean and the standard deviation of the classification accuracy from the cross-validation. Accuracy is computed as a number (probability) between 0 and 1.

From Tables 2 and 3 it is evident that the Classifier B outperforms Classifier A. But we should note that Classifier A uses only one instance of the Sequence for identification, whereas Classifier B uses multiple instances for a combined result. Classifier A will be suitable for applications like Continuous Authentication [5] which needs to authenticate a user immediately upon receiving a biometric sample. Note that Table 2 shows the accuracy for identification tasks (multiclass classification) rather than for verification (two-class classification with a claimed identity). We can in principle get even better performance for verification by choosing a sequence that works best for each person, i.e., since we know the identity of the person being verified, we can sacrifice universality for discriminability.

Table 2. Performance of Classifier A - English Words

Sequence	Mean of Accuracy	Std. dev. of Accuracy
FOR	0.0598	0.1224
TO	0.0838	0.1841
THE	0.0562	0.1504
YOU	0.0512	0.0733
IS	0.0538	0.0432
IN	0.0573	0.0669
AND	0.0878	0.2682
OF	0.0991	0.2809

Classifier B can be aptly called a post-login identifier, as it needs significant amount of keystroke data to identify the person. The identification accuracy is very high, but the price to pay is the large number of samples required. Although it might not be suitable for Continuous Verification, it can used as a forensic tool to identify the person after data collection. In our experiments, we also observed that the accuracy of Classifier B increases with the number of samples. We surmise that this is due to better estimation of the histogram.

3.2 Results II - For Non-English Sequences

In order to perform keystroke identification in a general setting, we cannot depend only on English words. Almost all users, even native English speakers, type abbreviated words every so often. For example, 'tmr' is frequently used to

Table 3. Performance of Classifier B - English Words

Sequence	Mean of Accuracy	Std. dev. of Accuracy
FOR	0.7955	0.2319
TO	0.9455	0.1184
THE	0.8409	0.2443
YOU	0.7364	0.2985
IS	0.9591	0.0796
IN	1.0000	0.0000
AND	0.8318	0.2191
OF	0.8000	0.1927

mean 'tomorrow'. In this age of short text messaging, such abbreviations are increasingly common. In fact, by regarding such words as coming from a foreign language, it is clear that our approach can be applied to other languages as well.

Running Classifier B on non-English Sequences produces Table 4. Generally the accuracies for non-English Sequences are higher than that for English words. Although the corpus listed 'THE' as the most frequently used word in English, it was no longer the case when non-English Sequences are considered.

4 Goodness Measure

Given that we are allowing non-English text, we can no longer rely on the Corpus to guide us in selecting useful sequences. How then do we select a Sequence for identification? We will need to look for them from the training data. According to Jain [13], a good biometric ought to satisfy seven criteria: Universality, Uniqueness, Permanance, Collectability, Performance, Acceptability, Circumvention. Of these, we only need to consider the first two, because the others have to do with technology, or user perception. Universality (commonality) means the biometric should be measurable across all users. Uniqueness (individuality) has to do with its discriminative power. From these two criteria, we derive a new *Goodness Measure* to measure the quality of a Sequence based on the criteria of accuracy, availablity and universality. First, a few definitions.

1. **Universality** (U)**.** A Sequence is not useful for identification unless it is commonly used by users. For English text, the Corpus lists 'THE' as frequently occuring because every person uses it. We define universality as,

$$U = \frac{\text{No. of Users having this Sequence in their keystrokes}}{\text{Total No. of users}} \tag{6}$$

2. **Accuracy** (A)**.** The classification accuracy of the Sequence. Note that $0 \leq A \leq 1$. Accuracy is a measure of how discriminative a Sequence is, i.e. its uniqueness.

3. **Expectancy** (E)**.** Unlike other kinds of biometrics, keystroke dynamics requires the system to wait until the user enters the required text. A particular string may be Universal, but the user may not type it frequently, thus keeping the system waiting for a long time. To capture this notion, we define the Expectancy of a Sequence to be the average number of keystrokes until an instance of the Sequence appears in the text. Intuitively, this measures how readily available the Sequence is. At best, the Sequence could appear on every keystroke (E=1); at worst, it might never appear in the text ($E = \infty$). For example, in the keystroke stream, 'TO BE OR NOT TO BE', E('TO') $= \frac{18}{2} = 9$ and E('THE') $= \infty$.

With the above definitions, we can now define Goodness Measure of a Sequence, G_m as follows:

$$G_m = \begin{cases} 0 & \text{if } E = \infty \\ \frac{U \times A}{E} & \text{otherwise} \end{cases} \tag{7}$$

Ideally, if all the factors are at their best (A=1,E=1,U=1), G_m will equal 1. In the Worst case (A=0 or E=∞ or U=0), G_m will equal 0. When $E = 1$, Equation (7) reduces to the special case,

$$G_m = U \times A \tag{8}$$

which may be interpreted as the Goodness Measure for a fixed-text keystroke identification system. Thus, our Goodness Measure is also applicable for traditional fixed-text systems.

Table 4 shows the Goodness Measure of a number of English and non-English sequences. The table is sorted by Accuracy, but a quick glance reveals that Accuracy does not mean a high G_m score. For example, the Sequence 'NE' (row four) has a high Accuracy but a low G_m score. The reason is its long Expectancy: one has to wait, on the average, over 400 keystrokes before this Sequence appears. For applications that require immediate authentication, 'NE' is a poor choice.

Table 4 also shows that the best performing English words (those that appear in Table 3) rank below non-English Sequences, both in terms of Accuracy and G_m score. Surprisingly, the Sequence 'THE' (third row from the bottowm) has a long Expectancy and is not Universal. This yields a low G_m score. We surmise that this counter-intuitive observation is because the subjects in our experiments did not write in complete, grammatical English. This, in turn, probably reflects the informal way English prose is used in everyday communication, rather than an indictment of the subjects' poor command of the language. Finally, the table also highlights the fact that Expectancy is the dominant criteria affecting G_m score. Many Sequences with approximately equal Accuracy and Universality scores differ greatly in their G_m scores because of different Expectancy values. For Keyboard Dynamics in a free-text setting, waiting for a long Expectancy Sequence limits how quickly the system can authenticate. Sequences with short Expectancy are more useful in this regard.

Table 4. Performance of Classifier B - non-English sequences

Sequence	Accuracy	Expectancy	Universality	G_m
AN	1.0000	113	1	0.008866
IN	1.0000	125	1	0.007986
NG	0.9955	150	1	0.006636
NE	0.9909	407	1	0.002433
LE	0.9864	294	1	0.003360
RE	0.9818	218	1	0.004498
TI	0.9818	324	1	0.003030
HE	0.9773	157	1	0.006226
EN	0.9773	207	1	0.004722
MA	0.9773	383	1	0.002549
ER	0.9727	245	1	0.003977
OU	0.9682	317	1	0.003051
IT	0.9682	339	1	0.002857
ING	0.9682	345	1	0.002802
AI	0.9636	312	1	0.003083
⋮	⋮	⋮	⋮	⋮
TO	0.9455	411	1	0.002298
THE	0.8409	350	0.95	0.002296
AND	0.8318	814	1	0.001022
FOR	0.7955	1062	1	0.000749

5 Conclusion and Future Work

In this paper, we presented a technique to identify a person based on Keystroke Dynamics in a general setting. This generalizes traditional fixed-text Keystroke Dynamics to free-text systems. Essentially, we identify a person based on a common list of fixed strings which we discover from analyzing users' keystroke logs. Our technique can also be used for verification, in which case each user can have his/her own list of strings.

We also found that non-English Sequences were more accurate than English words. This is useful because the prevalence of new communication technologies, such as instant messaging, online chat, text messaging, etc., means that users increasingly use informal English (containing abbreviations and even new words) when composing messages. This is true even for native English speakers. To guide our selection of good non-English words to use, we proposed a novel Goodness Measure based on well-studied properties that all biometrics ought to possess.

In the future, we would like to conduct the experiments on a larger pool of users to see if our results hold up. Also, we intend to investigate the effect of different keyboards on a person's typing speed, and how we may mitigate against this. Finally, it would be interesting to see if keystroke dynamics can distinguish between trained and untrained typists.

References

1. Shepherd, S.J.: Continuous authentication by analysis of keyboard typing characteristics. In: IEEE Conf. on Security and Detection, European Convention, pp. 111–114. IEEE Computer Society Press, Los Alamitos (1995)
2. Fry, E.B., Kress, J.E., Fountoukidis, D.L.: The Reading Teachers Book of Lists, 3rd edn.
3. Monrose, F., Reiter, M.K., Wetzel, S.: Password hardening based on keystroke dynamics. In: Proceedings of the 6th ACM Conference on Computer and Communications Security. ACM Press, New York (1999)
4. Rodrigues, R.N., Yared, G.F.G., Costa, C.R.D., Yabu Uti, J.B.T., Violaro, F., Ling, L.L.: Biometric Access Control Through Numerical Keyboards Based on Keystroke Dynamics. In: International Conference of Biometrics, pp. 640–646 (2006)
5. Kumar, S., Sim, T., Janakiraman, R., Zhang, S.: Using Continuous Biometric Verification to Protect Interactive Login Sessions. In: ACSAC, pp. 441–450 (2005)
6. Joyce, R., Gupta, G.: Identity authentication based on keystroke latencies. In: Communications of the ACM, vol. 33(2), pp. 168–176. ACM Press, New York (1990)
7. Villani, M., Tappert, C., Ngo, G., Simone, J., St. Fort, H., Cha, S.: Keystroke Biometric Recognition Studies on Long-Text Input under Ideal and Application-Oriented Conditions. In: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop, p. 39. IEEE Computer Society, Washington (2006)
8. Rao, B.: Continuous Keystroke Biometric System M.S. thesis, Media Arts and Technology, UCSB (2005)
9. Leggett, J., Williams, G., Usnick, M., Longnecker, M.: Dynamic identity verification via keystroke characteristics. Int. J. Man-Mach. Stud. 35(6), 859–870 (1991)
10. Obaidat, M.S., Sadoun, B.: Keystroke Dynamics Based Authentication. Ch. 10, Textbook (1998), `http://web.cse.msu.edu/~cse891/Sect601/textbook/10.pdf`
11. Duda, R., Hart, P., Stork, D.: Pattern Classification, 2nd edn. John Wiley and Sons, Chichester (2000)
12. Gunetti, D., Picardi, C.: Keystroke analysis of free text. ACM Transactions Information Systems Security 8(3), 312–347 (2005)
13. Jain, A.K.: Biometric recognition: how do I know who you are? In: Proceedings of the 12th IEEE Signal Processing and Communications Applications Conference. IEEE Computer Society Press, Los Alamitos (2004)