A Computer User Virtual Mark - Keystroke Dynamics using Binary and Multi-Class SVM

Itay Kirshenbaum 026559922
Mor Sela 032511636

April 27, 2011

Abstract

As an attempt to obtain a more practical approach to user authentication systems, with low cost and easy implementation we use a known keystroke dynamics dataset and the SVM mechanism to build a keystroke-aware system that allows differentiation between different registered users.

1 Introduction

1.1 Password Authentication

Password authentication is the leading method of user authentication nowadays. This method relies on the correct input of a user/password combination to gain entry. We use password authentication dozens of times a day - Sending email, logging on to a work computer, performing actions in a bank account, etc.
This method suffers from some serious disadvantages:

- Users are forced to memorize their password. This results, in most cases, one of three outcomes:
  - Deliberately choosing a weak and easy to remember password
  - Choosing and forgetting a strong and complex password
  - Writing the password down (Most commonly in an easy-to-find location)
- As a result of the above - Passwords are generally weak and easy to guess, especially given prior knowledge of the user
- The right user/password combination, may allow any (malicious) user to gain entry to any personal data.

The above sums up to a very weak security offered by this scheme as it almost makes the process of falsifying ones identity accessible to anyone.

The solution, it seems, lies in biometric authentication - methods for uniquely recognizing humans based upon one or more intrinsic physical (some particular structural characteristic such as hand size or iris format and color) or behavioral traits (some particular behavioral characteristic such as typing speed or writing pressure) inherent to a user. The main problem with the intrinsic physical methods is the costs and specialty equipment required to engage in them. as opposed to them behavioral typing information can be very convenient because usually no extra hardware is necessary. All the behavioral information can be obtained by software systems, what generally implies in lower cost than hardware development. Moreover, nothing changes in the way the user authenticates himself, what makes it more acceptable. We look at keystroke dynamics as a viable biometric method assisting (or replacing altogether) password authentication schemes.

1.2 Keystroke Dynamics

Keystroke dynamics deals with the manner and rhythm in which an individual types characters on a keyboard. The goal is to identify the unique keystroke features in order to extract a pattern later used for identification or authentication including: latency between consecutive keystrokes, flight time, dwell time, based on the key down/press/up events (as shown in figure 1), overall typing speed, frequency of errors (use of backspace) and control keys (use of left/right shift). Systems do not necessarily employ all these features; In our chosen data set only latencies and dwell time is used (one should notice that some of the before-mentioned features are a combination of the others).

Keystroke dynamics method is not without difficulties:

- A user logging on to the same website from several places may be using a different keyboard each time, effecting his typing rhythm.
- A user may type differently depending on his concentration levels (e.g. if he is watching TV at the same time)

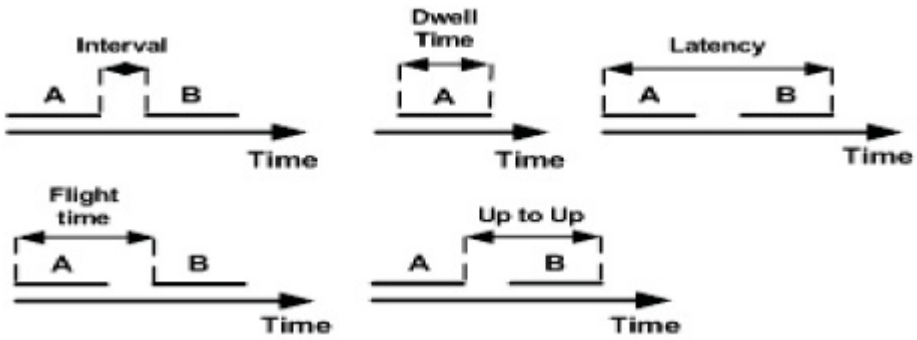

Figure 1: Keystroke metrics: latency, interval, dwell time and flight time

- A user may be pre-occupied in a way that affects his keyboard usage (Typing while talking on the phone may force the user to type with only one hand as opposed to both hands as he normally does)

Nevertheless, being non-intrusive and low cost, this method is quickly emerging as a strong alternative to traditional biometric methods. However, the great question that lies on this kind of authentication is how precisely can we verify the user typing behavioral characteristics. Many studies have shown that this kind of identification is viable but we are still far from obtaining satisfactory results.

The work on keystroke dynamics was initiated by R. Gaines, etc [5]. Afterwards much work was done on the subject, using many models and machine learning constructs, such as T-test, neural networks, autoregressive models and more.

As of now, no detector has achieved error rates that are satisfactory in repeated evaluation and so keystroke dynamics could not be deployed as a sole access-control technology. As a result, much of the work in the field concerns with comparing detector performances to achieve better results. Unfortunately, although researchers have conducted experiments to measure the performance of various detectors, these experimental results are impossible to compare. Many factors vary from one evaluation to another. Moreover, as each new paper had to come up with its own data a large portion of each research was concerned obtaining that data. That also has led to many data sets which were too small for practical testings.

For those reasons and more, Killourhy and Maxion [2] in their paper set up to create a standardized keystroke-dynamics data set; 51 subjects were given a 8 data-collection sessions (of 50 passwords each, where they waited between each sessions at least one day, to capture some of the day-to-day variation of each subject's typing), for a total of 400 password-typing samples. The typed password was the same for all of the subjects, and was chosen by a strong-password generator (.tie5Roanl). Out of the raw typing data the following features were extracted - keydown-keydown times, keyup-keydown times, and hold times for all keys in the password, including the enter at the end. For each password, 31 timing features were extracted and organized into a vector. This is the data which was used in the experiments in this paper.

Most of the times nowadays, keystroke dynamics is still regarded as a theoretical concept. Moreover, most of the systems proposed must behave as an anomaly detectors, in order to be able to differentiate any new user (or impostor). In practice we often desire a simple and robust authentication system (and not all-novel-impostor-resistant system). As opposed to most previous researches we do not aspire to obtain all the potential impostors data pattern (as this is practically impossible). Instead we propose a closed environment verification system in which a prior registration and user knowledge is demanded.

In the article by Killourhy and Mixan [2] the detector which was implemented for SVM was using a simple one-class SVM approach (as proposed by Yu and Cho [6]). SVM proves to be a solid algorithm and the article proposes

some options for how to extend the usage in the future, such as employing binary and multi-class classifiers. Our paper discusses implementing those binary and multi-class classifiers. of course, there are many options for how a binary classifier might be trained - to discriminate between two typists, or between one typist and a pool of typing data comprised of many other typists. The rest of the article will try to address the issues with the concept.

Section 2 of this paper presents the authentication method. The evaluation methodology and analysis of the authentication method is described in Section 3. Further experiments are introduced in Section 4. Finally, Section 5 summarizes the whol paper and discusses some future work.

2 Proposed Method

In order to solve the problem of novel impostors detection, support vector machine (SVM) was employed, mainly because of its capability of presenting consistent algorithm status. Specifically, we propose a solution employing two-class SVM which provides the training data with an overall coverage on the possible impostors. The algorithm of two class SVM constructs an optimal separating hyperplane that maximizes the margin (i.e. the distance between the positive and negative hyperplane). by mapping the input space into a higher dimensional feature space. This mapping is determined by a kernel function.

2.1 Registration

As the proposed system demands a known environment. In order to participate in the system, each new user must enter a variable amount of times its password (in our case the same password). That data is used in conjunction of all the other users data to rebuild the system models.

2.2 Verification

When a genuine user attempts to gain access he needs to enter the password a number of times; The system performs classification based on all the relevant models to verify that the user is inside the system (it can actually tell who the user is with no prior input). If the result user is legal within the system-data, the system may now verify that it is the same identification the genuine user tried to log on with (may be unneeded). If everything went through fine, the authentication succeeds.

When an impostor attempts to gain access using a known password, the system (with high probability) does not find any adequate user for the impostor which results unsuccessful log-on.

3 Evaluation Methodology

3.1 Training and Testing

In order to evaluate the error of the proposed method, we have created a scenario (not unlike [2]) in which a user's long time password has been compromised by an impostor. We measure how well does our system is able to discriminate between the impostor's typing and the genuine users typing in this scenario.

The SVM that was chosen for the project was nu-SVC (The choice was based on empirical testing of all the available SVM types and their success on parts of the data). The kernel function which was used is RBF (radial basis function) which is a commonly used kernel function:

$$K(x, y) = e^{-\gamma|x-y|^2}$$

The training phase was accompanied by a strict parameter selection using cross-validation for each of the classifier models.

1. The algorithm begins by training all of the binary classifiers (each binary classifiers distinguishes between a specific user and the rest of the users) so that for each user a model of the users typing behavior is trained using the timing vectors from the first 200 password repetitions typed by all of the users, where the typing for the chosen (genuine) user is marked as one class and all the other users (impostors) typing as the other class.

2. The user-test stage uses the remaining 200 repetitions of each chosen user. For each of the binary classifiers models we compute the prediction scores, recording those as user scores.

3. The impostor-test stage uses the first five repetitions typed by a each of the 51 impostors, and test against the real-user classifier with them, recording those as impostor scores.

It should be noted that in the user scores only the classifier for the chosen user is expected to declare that the given user is chosen user, all the other classifiers are expected to result "not my user" result. The impostor-test stage test against the real user classifier (as for each user we want to verify that no other user can imitate that user typings).

3.2 Results and analysis

To measure the model performance, we used two known performance metrics for biometric systems: The rate of authorized users wrongly authenticated as impostors - false rejection rate (FRR), and the rate of impostors wrongly authenticated as authorized users - false acceptance rate (FAR). Normally, any biometric verification system would like to minimize all of those metrics. In access control systems, the requirement is to keep the impostors out. Thus, false rejection is considered the least important error, while FAR is usually considered to be the most important error.

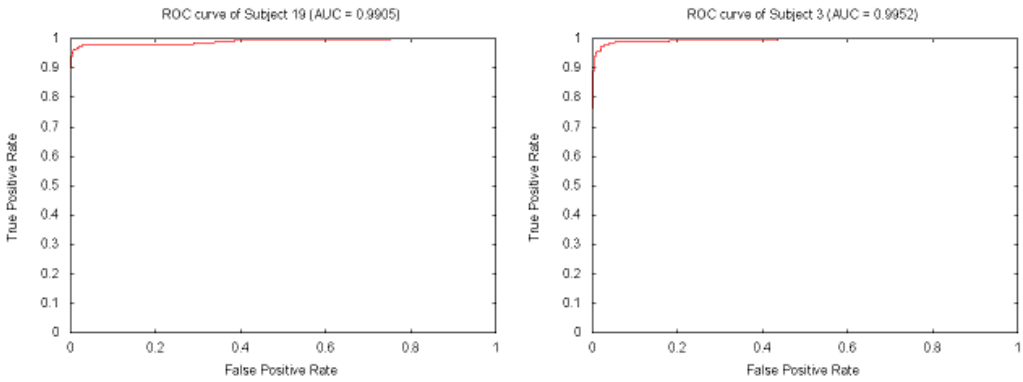

Figure 2: Example ROC curves for subject 19 (on the left) and subject 3 (on the right)

Another visualization used for the evaluation is the receiver operating characteristic (ROC) curves. The ROC plot is a visual characterization of the trade-off between the FAR and the FRR. In general, the matching algorithm performs a decision based on a threshold which determines how close to a template the input needs to be for it to be considered a match. If the threshold is reduced, there will be less false non-matches but more false accepts. Correspondingly, a higher threshold will reduce the FAR but increase the FRR. Proximity to the top-left corner of the graph is a visual measure of performance. An example for some of the ROC curve obtained is shown in Figure 2.

FRR and FAR were computed for each of the subjects and a portion of them can be seen in the table below:

Subject Number	**FRR**	**FAR**
3	0.004	0.06
12	0.09	0.003
15	0.385	0.005
18	0.285	0.001
21	0.705	0.002
27	0.525	0.003
30	0.125	0.001
36	0.14	0.012
43	0.08	0.016
53	0.04	0.015

Table 1: Binary Class SVM Overall Performance

Afterwards the averages of all the 51 subjects FRR and FAR were taken and resulted average FRR of 23.4% and FAR of 0.23%. As one expects the FRR of the system is bad (which is partially reasonable). We assume that a much lower FRR can be reached given a better tuning for the overall SVM constructs. The result FAR is quite striking considering the model - almost no impostors (using quite a big bank of impostors) seems to be able to log-on using any of the users typings.

4 Further Experiments

4.1 Multi-Class SVM

The second part involved referencing each participant as a whole different class and trying to find a multi-class SVM which performs well.

The parameters used were nu-SVC with nu of 0.072 and RBF kernel with gamma of 0.09. The process: For each user, half of the sessions were (randomly) selected as training data while the other half was selected as test data. The data was evaluated to obtain FAR and FRR (Per-user and global).

FAR was computed based on the user's attempts. FRR was computed based on the number of other users (impostors) which were able to identify themselves as the user. The process was repeated 10 times to average out the impact of the random selection.

The following table details the average FAR and FRR computed in each of the runs:

Iteration	**FRR**	**FAR**
1	0.00421	0.2149
2	0.00379	0.1933
3	0.00423	0.21549
4	0.00403	0.20559
5	0.00413	0.21078
6	0.00374	0.19059
7	0.00398	0.20284
8	0.00397	0.19745
9	0.00419	0.21363
10	0.00415	0.21167

Table 2: Multi-Class Overall Performance

With multi-class SVM, the resulted averages were FRR of 20.56% and FAR of 0.4%. While the FRR in this case is slightly better than in the binary case (23.4%) - The FAR is almost twice as much, but still remains in an acceptable range.

4.2 Password Selection

In attempting to test the robustness of the method, we tried playing with the features available to determine minimal requirements for the password. How well would the method do with only partial data? How is the performance impacted if a weaker password is chosen - for example only 5 characters? Would it have any success at all looking at only a single character?

The following table summarizes the results of our experiment:

Password length	**FRR**	**FAR**
10	0.004	0.2056
8	0.00478	0.24392
5	0.00744	0.3792
3	1.00094	0.55794
1	1.00582	0.807

Table 3: Various Password Length Performance

The direct link between the FRR and FAR is clear from the above results. In any way, password selection still remains an important part of the authentication scheme, as simple and short passwords drive the error rates up.

5 Conclusions

In this paper, binary class SVM was employed to detect and verify genuine users in a closed-verification system. As opposed to many other constructs SVM has a low computational cost with adequate performance. As there is always a trade-off between FRR and FAR we tried to address the more concerning metric in our eyes - FAR, as in an authentication system in a real world application there are millions of potential impostors. Comparatively, binary class SVM seems to perform less efficiently than detectors suggested by [2]. One of the suggested reasons for this is that there are overlaps between the normal data patterns and outliers. Moreover, we have had a lot of variation with our results (for good and for worse) meaning that better parametrization should result better results.

Multi-class SVM was also tested, assuming that in a close group situation it should perform at least as well as the binary case. We were proven wrong as the FAR was at the same weak levels and the FRR was slightly worse. It was also evident that the addition of more subject into the group would hurt the performance. In essence, we have learned there is no real reason to use this variant of SVM in this situation, supporting the results in [2].

The results also stress the importance of a policy enforcing strong password selection. While this fact still keeps long and complex passwords in play, the

advantage to keystroke dynamics is that passwords can be written down and kept in an obvious place without fear. Even a unified password selection for all users would not compromise the level of security, and would allow a smart and well planned selection.

5.1 Future Work

- Fine Tuning for the SVM constructs - SVM has a lot of inner strength, what we did in this paper can surely be amplified using not a lot extra work, using better parameterization and better use of the overall construct.
- Large scale databases
  - Most organizations support hundreds or thousands of users
  - Web services support millions
  - Interesting to see if the SVM method (or any other method) scale up nicely
- Attacking keystroke dynamics - Attempting a brute force computerized method of playing with the timings of entering the password in order to break in
  - May be hard for only 50 subjects
  - Would be much easier on large company / free web service with thousands or millions of users (A random selection may fit one of the many users)
  - How long would it take to brute force this data set?
  - Identifying as any user vs identifying as a specific user
- Assessing strength of passwords
  - Generating similar databases, each with a different password
  - Identifying the features that lead to the best FAR / FRR
  - Defining a policy for selection of strong passwords
- Free text identification
  - As training samples, each user would be asked to type a pre-defined block of text
  - The authentication process would require the user to type a randomly selected sentence
  - Selection of the correct training text is crucial
  - Is free text a viable option?

References

[1] Moskovitch, R., Feher, C., Messerman, A., Kirschnick, N., Mustafic, T., C,amtepe, A.A., Lo"hlein, B., Heister, U., Mo"ller, S., Rokach, L., and Elovici, Y. *Identity Theft, Computers and Behavioral Biometrics*. In Proceedings of ISI. 2009, 155-160.

[2] Kevin S. Killourhy and Roy A. Maxion, *Comparing Anomaly Detectors for Keystroke Dynamics*. in Proceedings of the 39th Annual International Conference on Dependable Systems and Networks (DSN-2009), pages 125-134, Estoril, Lisbon, Portugal, June 29-July 2, 2009. IEEE Computer Society Press, Los Alamitos, California, 2009.

[3] Sang, Y., Shen, H. and Fan, P. *Novel Impostors Detection in Keystroke Dynamics by Support Vector Machine*. LNCS Springer Berlin/Heidelberg, pp.666669, ISSN 0302-9743, 2004.

[4] de Oliveira, M., Kinto, V.S.E., Hernandez, E.D.M. and de Carvalho, T.C. *User Authentication based on Human Typing Pattern with Artificial Neural Networks and Support Vector Machine*. SBC, 2005.

[5] Gaines, R., Lisowski,W., Press, S., Shapiro N., *Authentication by keystroke timing: some preliminary results*. Rand Report R-256-NSF. Rand Corporation. 1980.

[6] E. Yu and S. Cho *GA SVM wrapper ensemble for keystroke dynamics authentication*. IAPR International Conference on Biometrics, 2006.

[7] F. Monrose and A. Rubin *Keystroke Dynamics as a Biometric for Authentication*. Future Generation Computer Systems, 16, 351-359, 2000.