Finite Context Modeling of Keystroke Dynamics in Free Text

Nahuel González, Enrique P. Calot

Laboratorio de Sistemas de Información Avanzados
Facultad de Ingeniería, Universidad de Buenos Aires
Ciudad Autónoma de Buenos Aires, Argentina
ngonzalez@lsia.fi.uba.ar, ecalot@lsia.fi.uba.ar

Abstract: Keystroke dynamics analysis has been applied successfully to password or fixed short texts verification as a means to reduce their inherent security limitations, because their length and the fact of being typed often makes their characteristic timings fairly stable. On the other hand, free text analysis has been neglected until recent years due to the inherent difficulties of dealing with short term behavioral noise and long term effects over the typing rhythm. In this paper we examine finite context modeling of keystroke dynamics in free text and report promising results for user verification over an extensive data set collected from a real world environment outside the laboratory setting that we make publicly available.

1 Introduction

Keystroke dynamics modeling has been applied to password verification as a means to reduce their inherent security limitations. Though its classification performance might not reach those of other biometric schemes making it unsuitable for international security standards, some reported results are outstanding considering how noisy the source data can be and how little extra effort is required for deployment.
Passwords and fixed short texts fit well the general framework of keystroke dynamics; being short sequences typed in a row and repeated often, their characteristic timings tend to be fairly stable in a broad sense. Free text does not enjoy the same privileges. Typing errors, corrections, misspellings, interruptions, pauses to think and attention lapses which are impossible to predict poison the source timing data. What is more, short term variations due to daily tiredness, stress or emotional shifts and long term effects of health and typing skills reeducation are strictly unavoidable. Due to the mentioned difficulties, the problem of free text analysis of keystroke dynamics has only began to be attacked recently [GP05] and in rather controlled conditions. In this paper[1] we examine finite context modeling of keystroke dynamics in free text and report promising results. The general framework we used admits many implementation parameters and selection strategies; their effects on classification performance are studied.

[1]The full length version of this paper can be found at http://lsia.fi.uba.ar/papers/gonzalez15.pdf

2 Theoretical considerations and common techniques

The most commonly used characteristic parameters are hold time -latency between key press and release-, wait time -latency between key release and next key press- and flight time -latency between successive key press events-; to a lesser extent, average typing speed and probability of error or usage frequency of backspace and delete. Usually, timing between consecutive keys -called digraphs- are used but occasionally latencies of bigger groups are chosen [BGP02].
Verification of static passwords, usually in the range of eight to twenty characters long, starts by forming a characteristic vector containing the sequence of values for one or more of the aforementioned parameters, measured during the password entry. The characteristic vector is compared with a pattern vector which is the result of a training process where the same password is entered multiple times. The pattern vector generally contains information on the sample mean, variance and eventually the shape of the adjustment function for all the parameters measured at each keystroke of the password.
Almost every technique for classification and machine learning has been tried for the analysis of keystroke dynamics, ranging from simple metric spaces and k-NN to state of the art classifiers like SVM or random forests; artificial neural networks, fuzzy logic and genetic algorithms have been tried too, with results not as promising. Killourhy and Maxion's review [KM09] has become a classic; an updated one can be found at [KAK11].
Two simple metrics are generally used to grade the quality of biometric systems, including keystroke dynamics modeling: *false acceptance rate* and *false rejection rate*. When a single metric is required for comparison, the *equal error rate* is preferred. Different reporting methodologies in the literature have hindered the cross comparison of results, a problem which is worsened by the diversity of acquisition protocols, data set size, depth or time span, and implementation parameters like minimum required observations for training or classifier threshold. See [GEAHR11] for a detailed review and methodological critique of past studies. We emphasize the lack of public real world data for evaluation, as opposed to that captured in a laboratory setting that artificially diminishes expected variations in the typing rhythm.
Though some isolated attempts were made at tackling its problems, keystroke dynamics analysis of free text has been elusive until recent years [MMCA11]. An original distance metric [GP05] using the degree of disorder of a latencies vector for n-graphs has shown promising classification performance and high tolerance to variations in typing rhythms.

3 Finite context modeling

The motivation behind the usage of finite contexts for modeling keystroke dynamics is based on the fact that prediction by partial matching [CW84] is one the best performing schemes -and asymptotically optimal- for natural language compression. Given that continuous use of a terminal involves extended usage of natural or artificial languages, a similar approach can be expected to be suitable to predict characteristic parameters of keystroke dynamics.

3.1 General modeling framework

A partition P is a sequence $k_0...k_m$ of key identifiers, codified in a hardware and software independent way to avoid variations due to differences in regional configuration if the sessions are captured in different terminals; a session or input text T is a sequence of partitions and a training text is a sequence of sessions. The n-order finite context C_j^n of the j-th key k_j -called leading- in a partition P is the sequence $k_{j-n}k_{j-n+1}...k_{j-2}k_{j-1}$ of n preceding key identifiers. For each characteristic parameter p -hold time, wait time, flight time, applied force or possibly others-, each context C that appears in the training set of a user u and its leading key k, a keystroke model $M_u(p, C, k)$ is created. The set $\mathcal{U}$ of all models M_u is called the user model. The structure of specific keystroke models is open and implementation dependent, being able to provide to the classifier meaningful statistical information about the sampled distribution of timing data for the parameter, context and key. Actually we used models that provided sampled mean and standard deviation, as well as being able to detect outliers using tail probabilities without assuming an underlying normal distribution. However, testing a different set of classifiers or refining techniques might require additional capabilities from the models.

3.2 Partitioning and filtering

As the full session content is not generally typed continuously but includes arbitrary pauses, it is split in partitions before being fed to the trainer or classifier with the purpose of restarting the contexts to zero length. A combined strategy was used to partition the text whenever the flight time exceeded a certain fixed threshold or three times the simple moving average for the previous flight times in the partition. Manual inspection showed the criteria matched partitions to humanly perceived pauses in typing; testing more complex strategies did not improve classification metrics. It was found that considering timing data of certain special keys -including modifiers (shift, ctrl, alt), navigation (arrows, page up, page down) and correctors (backspace and delete)- worsened the classifier's performance even though their consideration in the contexts of the next keystrokes improved it. Consequently, their timings but not the keystrokes themselves are removed from the characteristic vector of the session being classified together with the digraph delays of the next keystroke.

3.3 Pattern vector reconstruction and classification

It can be assumed in general that not enough repetitions of the input text -free and thus unpredictable- will be found in the training set, if it can be found at all. Thus, to feed a pattern vector to the classifier so it can be compared with the characteristic timing vector of the input text, the former must be reconstructed from the keystroke models in $\mathcal{U}$. For every keystroke k_i in the input text and every characteristic parameter p, one of the models

$M_u(p, C, k_j)$ is chosen for C in C_j^0 to C_j^j. The best strategy is not necessarily the obvious one of picking the longest available context; the order of the optimal predicting context can vary dynamically and, surprisingly, can be much shorter than expected. Also, not all the parameters p might be meaningful at all times and some can be excluded from the recreated pattern vector (i.e. the first key in every partition can only have an empty context and a single meaningful parameter, hold time). Using an expected vector pattern rebuilt from fragments of training texts does not introduce explicit constraints to classifiers like distance metrics and outlier counting. Unluckily, adapting state of the art classifiers like random forests or SVM to the task does not seem trivial. For this experiment we tested only the aforementioned; a clever scheme using the latter ones can probably improve over our results.

3.4 Experimental setup

The data set for the evaluation of the proposed method contains 17158 sessions from 146 users spanning a five months period; the sessions contain up to 10013 keystrokes and an average of 743 keystrokes. Intervals between typing sessions of a certain user can range from hours to days. Sessions were not collected in a special purpose or laboratory setting but were live recorded from daily work with written consent of those involved and their employers; thus, noise factors affecting typing cadences such as interruptions, attention lapses and short and long term emotional, mood or health variations of the individuals are not excluded. Different keyboards with different regional configurations were used, even by the same individuals. The collecting application whose original purpose was to create written reports of professional activities was modified to record keystroke timing; the content does not include personal data or activity trails of the participants. Users' positions are not computer related except for the mentioned purpose of report writing and their typing skills vary from mediocre to excellent, without single finger typists. For each session the timing of key down and key up events were recorded with a precision of 1 millisecond and the presumed user identity, previously authenticated with a passphrase, is included.
In order to encourage collaboration, independent verification of the presented results and further research of keystroke dynamics in realistic scenarios, we make the data set publicly available at the laboratory website and promise to update it as it grows. While most public data sets focus on repetitions of similar strings by the same or different users, we are not aware at the moment of writing this article of any other with this size consisting purely of extended free text without repetitions, except [MMCA11] which is about one third the size of ours but spans a longer period of twelve months.

4 Implementation and results

All users with at least 200 sessions were selected for evaluation while the rest were kept only as impostors; 18 users had enough sessions to be evaluated. For every legitimate

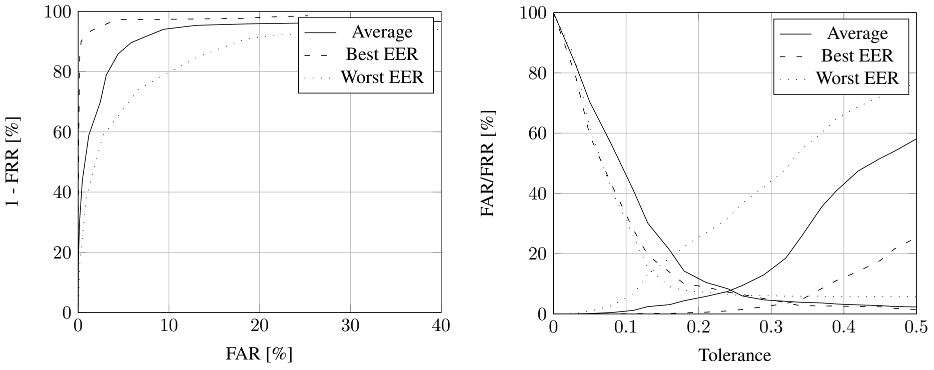

Figure 1: Detailed classifier performance. *Left*, left side of the ROC curve. *Right*, FAR and FRR versus tolerance. Best, worst and average user results for the best strategies combination.

user under evaluation, their first 150 sessions were used as the initial training set to build their keystroke dynamics profile; the remaining sessions where used as challenges, both legitimate and fake, in the same chronological order in which they were captured. After recording the classification result from a legitimate session, its content was used to update the user model.

Due to computational power constraints, a maximum context order of 6 was used; in spite of this limitation, the optimum classification performance seems to be reached around an order of 4. Using less than 100 sessions for the initial training degrades the performance of the method rather fast so a reasonable margin of 150 was chosen; identically, a minimum of 50 legitimate evaluation sessions was used to avoid meaningless variations in the false rejection rate if just a few were misclassified. The observed parameters were hold time, wait time and flight time.

Following [Kil12] we do not consider a certain method to have an EER pointwise defined but randomly distributed; thus, results are reported as average EER together with their standard deviation, maximum and minimum values for the considered users. The best results, with an average EER of 7.56% -minimum of 3.49%, maximum of 13.58%- and a standard deviation of 3.58 were obtained with the combination of choosing the longest available context as model selection strategy, updating models with an exponential moving average, requiring only 10 observations before considering a model valid and estimating the best classifier after 100 evaluation sessions. A detailed plot of FAR and FRR versus tolerance (calculated as fraction of maximum allowed distance or outlier count percentage threshold before rejection) and ROC curves for the best, worst and average user results is shown in figure 1. The results for other combinations of strategies mentioned in this section are shown in table 1.

Three averaging strategies were tested to assess the impact on classification metrics of model updating in addition to a base case where models were not updated after initial training: averaging over all past observations, simple ($n = 50$) and exponential ($\alpha = 0.98$) moving averages. EERs by maximum context order are shown in figure 2. As expected, updating strategies beat the static base case and their performance improves with the importance given to recent observations, making the exponential moving average the best.

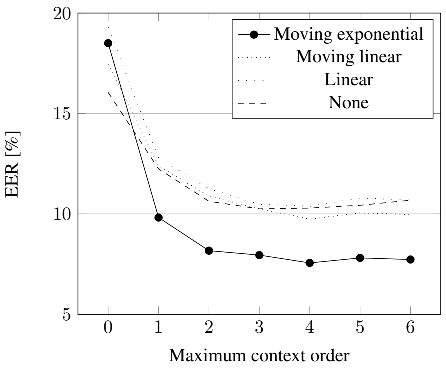

Figure 2: Comparison of model updating and averaging strategies.

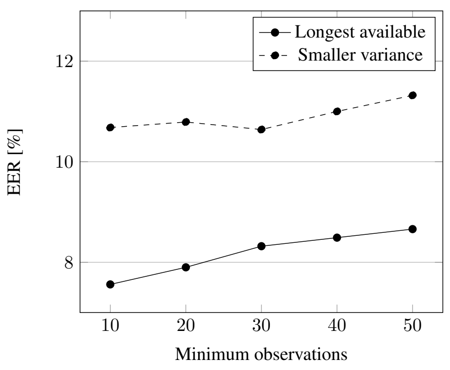

Figure 3: Effect of minimum required model observations on EER for both update strategies.

The minimum required observations to consider a certain model valid is an implementation parameter with influence over the classification performance not only due to the assumed convergence speed of the models but also because of the early availability of higher order models in the classification process, and their updating speed. It is generally claimed that a minimum of 20 observations is required because of the normality assumptions of the underlying variables; however, figure 3 shows that as little as 10 observations are enough to reach optimum or almost optimum EER values with the context selection strategies evaluated next and that increasing their minimum number is slightly but steadily harmful.
A comparison of two context selection strategies shows that the simplest one, selecting for every key k_j the longest available context, outperforms for every order the strategy of choosing from C_j^0 to C_j^{MAX} the one that gives the model with the smallest variance.
As expected, the choice of classifier has a noticeable effect on performance. The average results by maximum context order are shown in figure 4 for euclidean distance, outlier count and Manhattan distance, which are named in ascending order of achievement; longest available context and 10 minimum required observations per model were used. It has been shown [Kil12] that the classifier which consistently gives best results is user dependent and not necessarily the one with the best average. Surprisingly, this phenomenon can be exploited with ease to improve classification metrics by splitting the evaluation set and using some sessions -ignoring their results not to bias the reports- for a second training phase that estimates which classifier will outperform the others for that user. We used 50 legitimate sessions and 50 impostor ones for that purpose; even though the best classifier for a certain user was not always chosen, selecting the classifier on a per user basis outperformed every single one on average.
Looking for predictors of classifier performance, it was found that the average of the sample standard deviation for all models is highly correlated to the first. In figure 5 the EER of each user for the best performing scheme is plotted against the predictor, together with a simple regression line. Different approaches like averaging only over the standard deviation of zero order models and over the mean hold time, flight time or both, also show some correlation but not as clearly meaningful as the initial approach.

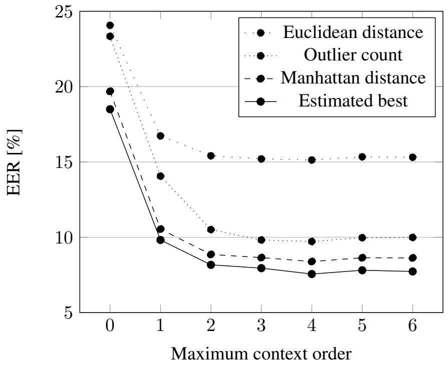

Figure 4: Comparison of basic classifiers and estimated best on a per user basis.

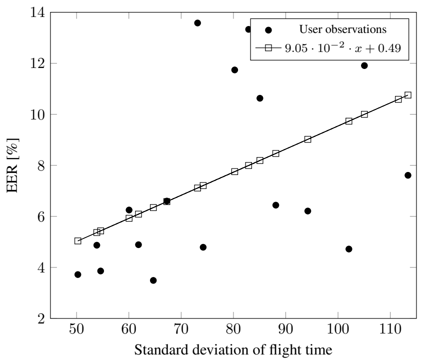

Figure 5: Regression of flight time standard deviation as a predictor of classification performance.

Observations	Order	Context	Classifier	Average EER	St. Dev.	Min	Max
≥ 10	≤ 4	Longest	Estimated best	7.56%	3.58	3.49%	13.58%
≥ 10	≤ 6	Longest	Estimated best	7.73%	3.89	3.26%	14.11%
≥ 10	≤ 6	Longest	Manhattan	8.63%	3.65	3.58%	14.29%
≥ 50	≤ 6	Longest	Estimated best	8.66%	4.79	2.93%	16.17%
≥ 10	≤ 6	Longest	Outlier count	9.99%	6.25	3.26%	23.76%
≥ 10	≤ 6	Min. var.	Estimated best	10.64%	5.7	3.36%	20.40%
≥ 10	≤ 6	Longest	Euclidean	15.31%	6.47	7.05%	26.14%

Table 1: Ranking of results

The general framework presented in this article admits extensions that were not evaluated and some questions remain unanswered. Can a smarter parameter weighting scheme improve classification performance? Are there better context selection or weighting strategies than those tested? Obviously, state of the art classifiers can beat distance metrics, but how to adapt them to the current scheme is not evident. Particularly, the unexpected results of outlier counts, which surpass euclidean distance -an excellent ranker for static passwords-, points to the fact that specific features of a reduced set of keystrokes might be more important than average deviation from a trained pattern vector. Feature extraction has already proved useful for static passwords before [YC03] and could scale well for free text. Consideration of additional biometrics characteristics which can be successfully extracted from the data with more confidence than identity (i.e. handedness or special key usage patterns) could boost performance of classifiers like random forests.
Finally, as the average of the sample standard deviation for all models is highly correlated with performance, we think the main path to improve the latter is identifying the sources of noise and removing their correlation from the main data. Stress levels, at least, seem to drift key latencies predictably [VZS09]; decorrelating the short term variation might reduce classification errors.

5 Conclusion

In this study we have shown the feasibility of user identity verification through keystroke dynamics analysis of free texts captured in a noisy real world environment; the data set was made public to encourage further research in the topic. The performance cannot compete with other biometric systems but has the advantage of being completely transparent and not requiring additional hardware or user actions; however, considering the additional difficulties in comparison with static password verification, our method seems promising and still has room for further improvements. Adapting models to short and long term variations in typing rhythms has proven critical to improve performance, as much as considering individual users as particular sources the classifiers need to adapt to. On the other hand, uncertainties in individual typing rhythms have been shown to be excellent predictors of performance and, as such, the ultimate limit to improvements. We consider this fact proves that keystroke dynamics, not only in fixed short texts but also during unconstrained typing, are rather unique for each user and that finite context modeling captures its essential features. Thus, our future research lines will focus on identifying noise sources of the timing characteristics to mitigate their negative effects on classification.

References

[BGP02] F. Bergadano, D. Gunetti, and C. Picardi. User authentication through keystroke dynamics. *ACM Transactions on Information and System Security (TISSEC)*, 5(4):367–397, 2002.

[CW84] John G. Cleary and Ian Witten. Data compression using adaptive coding and partial string matching. *Communications, IEEE Transactions on*, 32(4):396–402, 1984.

[GEAHR11] R. Giot, M. El-Abed, B. Hemery, and C. Rosenberger. Unconstrained keystroke dynamics authentication with shared secret. *Computers & security*, 30(6):427–445, 2011.

[GP05] Daniele Gunetti and Claudia Picardi. Keystroke analysis of free text. *ACM Transactions on Information and System Security (TISSEC)*, 8(3):312–347, 2005.

[KAK11] M. Karnan, M. Akila, and N. Krishnaraj. Biometric personal authentication using keystroke dynamics: A review. *Applied Soft Computing*, 11(2):1565–1573, 2011.

[Kil12] Kevin S Killourhy. A scientific understanding of keystroke dynamics. Technical report, DTIC Document, 2012.

[KM09] Kevin S Killourhy and Roy A Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*, pages 125–134. IEEE, 2009.

[MMCA11] Arik Messerman, Tarik Mustafic, Seyit Ahmet Camtepe, and Sahin Albayrak. Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics. In *Biometrics (IJCB), 2011 International Joint Conference on*, pages 1–8. IEEE, 2011.

[VZS09] Lisa M. Vizer, Lina Zhou, and Andrew Sears. Automated stress detection using keystroke and linguistic features: An exploratory study. *International Journal of Human-Computer Studies*, 67(10):870–886, 2009.

[YC03] Enzhe Yu and Sungzoon Cho. GA-SVM wrapper approach for feature subset selection in keystroke dynamics identity verification. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 3, pages 2253–2257. IEEE, 2003.