Continuous Authentication using Mouse Dynamics

Soumik Mondal and Patrick Bours

NISlab, Gjøvik University College
Postboks 191, 2820, Gjøvik, Norway
soumik.mondal@hig.no
patrick.bours@hig.no

Abstract: In this paper, we demonstrate a new way to perform continuous authentication using Mouse Dynamics as the behavioural biometric modality. In the proposed scheme, the user will be authenticated per mouse event performed on his/her system. We have used a publicly available mouse dynamics dataset and extracted per event features suitable for the proposed scheme. In this research, we have used the mouse dynamics data of 49 users and evaluated the system performance with 6 machine learning algorithms. In this approach, the genuine user has never been classified as an impostor throughout a full session whereas the average number of mouse actions an impostor could perform before detection is 94 from the best classification algorithm with a person based threshold.

1 Introduction

For most existing computer systems, once the user identity is verified at login, the system resources are available to that user until he/she exits the system or locks the session. In fact, the system resources are available to any user during that period. This may be appropriate for low-security environments, but can lead to session hijacking, in which an attacker targets an open session. In high risk environments or where the cost of unauthorized use of a computer is high, continuous verification/authentication of the user's identity is extremely important. Continuous Biometric Authentication Schemes (CBAS) has built around the biometrics supplied by the user behavioural characteristics and continuously check the identity of the user throughout a session [Bou12].

In this research, we are going to evaluate the Mouse Dynamics Biometric modality for Continuous Authentication. Mouse Dynamics is the way a user is interacting with his/her system through the mouse. Similar to Keystroke Dynamics [add references] does Mouse Dynamics (MD) not require any special hardware to capture the biometric data. From 2003, has MD become an interesting research topic in the area of behavioural biometrics due to its non-intrusiveness and convenience [EM03, GF03] for the user.

We found that the current research on continuous authentication reports the results in terms of EER or FAR and FRR over either the whole test set or over chunks of a large, fixed number of events. This is then in fact no longer continuous authentication, but at best periodic authentication. In this paper we focus on actual continuous authentication that

reacts on every single mouse action from a user. The contribution in this paper is the following:

- New scheme for continuous authentication;
- Verifying the genuineness of the user for every mouse event;
- Use 6 different classification algorithms to measure the system performance; and
- Modification of the trust model from [Bou12].

The remainder of this paper will be as follows. Section 2 will describe some of the research done in this area. In Section 3, we will discuss some basic knowledge about the machine learning algorithms which we have used in this research. We will discuss the dataset and the feature extraction process in Section 4. In Section 5 the trust model will be described. Section 6 will analyse the results obtained in this research, while Section 7 presents the results of our analysis method on the used dataset. Finally, in Section 8 we will conclude this research with future work.

2 State of the Art

Gamboa et al. [GF03, GF04] considered mouse strokes for continuous authentication. Each stroke was characterized by a 63-dimensional feature vector including spatial parameters such as angle and curvature, and temporal parameters such as velocity and acceleration. They used the data of 50 users and report that the EER is the function of iteration time and the number of strokes per user. They have reported results on 1 stroke (EER of 48.9%), 50 strokes (EER of 2%) and 100 strokes (EER of 0.7%).

Zheng et al. [ZPW11] used angle-based metrics of mouse movements for user verification. They have used 30 users (different ages, educational backgrounds, and occupations) and SVM as a classification tool. They have reported an EER of 1.3% with the requirement of 20 mouse clicks.

Pusara et al. [PB04] build a continuous authentication system using mouse movements and mouse events as features. They performed an experiment with 18 users (on average 2 hours of data per user) and they used Decision Tree Classifier with smoothing filters for classification. They reported an FRR of 0.43% and a FAR of 1.75% with the verification time ranging from 1 minute to 15 minutes.

Ahmed et al. [AT07, NTA10] build a continuous authentication mechanism with mouse dynamics. They collected data of 49 users and achieved a FAR of 0% and an FRR of 0.36%. Their dataset is publicly available and was also used in this research. They used fuzzy classification based on the learning algorithm for Multivariate Data Analysis and used a score-level fusion scheme to merge corresponding biometric scores.

Feher et al. [FEM+12] used individual mouse actions (contrary to using a histogram over a number of mouse actions) as a feature for continuous authentication. They have used 25

volunteers (21 male and 4 female) to collect data in their experiment and used Random Forest Classifier for data analysis. They have achieved an EER of 1.01% (for 30 actions) with the authentication time of less than 2 minutes.

Lin et al. [LCL12] build a continuous authentication system by using everyday mouse interaction data on a windows computer. They have used data of 11 volunteers and created 3 sample sets. Set A contained the feature vectors of the mouse movements for the complete file-related operations in Windows Explorer. For comparison, set B contained the feature vectors of the mouse movements for operating Windows Explorer, and set C contained the feature vectors of the mouse movements for operating the computer. Their best results were obtained with data set A as was to be expected.

Schulz [Sch06] has claimed a continuous authentication system using mouse dynamics. They collected in his experiment, data of 72 users. He has used three features (1) Length and number of a movement sample of the mouse curve; (2) Curvature and inflection; and (3) Curve straightness characteristics. Furthermore did they use Euclidean distance for classification. They achieved an EER of 24.3% when using a sample size of 60 mouse curves. This performance improved to 11.2% when using a sample size of 3600 mouse curves.

Shen et al. [SCG12] designed a mouse interaction based continuous authentication system. In their system, there is no need for impostor training data. They build their system based on the data of 28 users focusing on different mouse events, mouse operations and mouse behaviour patterns and used different classifier for classification. The best result was a FAR of 0.37% and an FRR of 1.12%, obtained using a One Class SVM detector. The authors also showed the effect of the length of a session. In particular for a session of 5 minutes they reported a FAR of 7.78% and an FRR of 9.45% while for a sessions of 10 minutes these values dropped to a FAR of 2.75% and an FRR of 3.39%.

3 Background Knowledge

For our analysis we tested various different machine learning algorithms. These were Naive Bayes Classifier, k-Nearest Neighbour (k-NN), Decision Tree Learning, Multilayer Perceptron, Radial Basis Function Network, and Support Vector Machine (SVM). We used the WEKA [HFH+09] and the LibSVM [CL11] software tools for the analysis of our data. Initial testing showed that SVM was the only method that always detected the impostors. For the other methods the probability that impostors were not detected ranged from 12% for k-NN to 76% for Decision Tree Learning. For continuous authentication it is obviously of the highest importance that impostors are detected and for this reason we present in the remainder of this paper only the results obtained with SVM. Some details of SVM are presented below.

3.1 Support Vector Machine

Support Vector Machine (SVM) is a very well-known supervised learning model which can be used for classification and regression analysis. This model is capable of creating a decision margin that is as wide as possible, depending on the Support Vectors (SV). The SV are those data points from the different classes that are closest to the decision line. In this research, we have used the LibSVM software distribution for the SVM classifier [CL11]. Initially we tried SVM with a linear kernel, but found that the classifier did not perform well due to the small feature set (see Section 4.2). We decided to use Gaussian kernel as a similarity measure function in this research.

4 Data Description and Feature Extraction

In this section, we are going to describe the dataset that we have used in this research. In Section 4.2 we describe the features that are extracted from the raw data for the analysis we perform.

4.1 Raw Data Description

We have used a publicly available mouse dynamics dataset [AT07, NTA10]. This dataset contains the mouse dynamics data collected from 49 volunteers. The volunteers were asked to use their computer and mouse in a normal, everyday fashion, without any restrictions on the tasks they had to perform. The data collection software stored the following 4 features for each mouse action from a volunteer:

- Type of action (1: Mouse Move; 2: Silence; 3: Point and Click; or 4: Drag and drop);
- Travelled distance in pixels;
- Elapsed time (with a 0.25 second accuracy);
- Direction of movement (a value between 1 and 8 according to the movement of the mouse. See Figure 1 for which direction corresponded to which value).

4.2 Feature Extraction

In [AT07, NTA10], as well as in many other works on continuous authentication, are statistical features extracted from the raw data. In our scheme we want to verify the identity of the user from every single mouse action. Therefor we cannot use statistical features

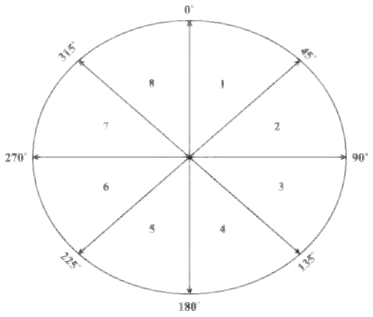

Figure 1: Direction of the Movements.

derived from the raw data, but are we looking for single event based features. We have extracted the following 5 features from the raw data:

Type of action We have explicitly removed the ”Silence event” actions from the raw data because we want to focus on the behaviour of the user. We therefor only used the other actions that were recorded, i.e. Mouse Move (MM), Point and Click (PC), and Drag and Drop (DD).

Direction of movement Taken directly from the raw data.

Speed of the mouse movement This equals the Travelled distance in pixels / the Elapsed time.

Reciprocal Acceleration of the mouse movement Equal to the Elapsed time / Speed of the mouse movement. We first tried to use the Acceleration of the mouse movement, but we got much better results when using the reciprocal of the acceleration.

Travelled distance in bins We did not use the travelled distance in its raw form, but decided to use a limited number of bins for the travelled distance range. The first 20 bins contained a 50 pixel range each, for example if the travelled distance was between 1 and 50 pixels then we assigned bin 1, if the travelled distance was within 51-100 pixels we have assigned bin 2 and so on. After that the bins grew in range, in particular we used 38 bins according to the following schedule:

- From 1 to 1000 pixels: Bin size is 50 pixels, so 20 bins in total;
- From 1001 to 2000 pixels: Bin size is 100 pixels, so 10 bins in total;
- From 2001 to 3000 pixels: Bin size is 200 pixels, so 5 bins in total;
- From 3001 to 4000 pixels: Bin size is 500 pixels, so 2 bins in total;
- More than 4001 pixels: Treated as a separate bin.

4.3 Data Separation

For each of the 49 users has the data set been split in a training set and a test set. The total training set was combination of the genuine user's training set and some impostor user's training sets. To avoid classifier biasness is the number of samples from the impostor users in the total training set is equal to the number of samples from the genuine user. The genuine user's part of the total training set was randomly chosen from the genuine user data. The amount of training data from the genuine user was 50% of his total data set. The imposer user's part was chosen at random from the other 48 users' data.

The total test set consisted of users' data samples that were not used in the training process. The amount of data used from the genuine user for testing equals 50% of his total data set.

5 Trust Model

Bours [Bou12] has described a trust model concept for continuous authentication using keystroke dynamics. He demonstrated that the trust level will increase or decrease according to the distance between the template and the current typing. In this research, we have used the classifier score (that is the probability of the genuineness of that event) to increase or decrease the trust value (denoted by C). We have used the distribution of the classifier score for the impostor users and the genuine user to adjust the parameter of the trust model. Figure 2 shows an example of the box plots for the distribution of the classifier scores for a randomly chosen genuine user (left) and a randomly chosen impostor user (right) when compared to the given genuine user. We can see that the median score for the genuine user lies above 0.5. On the other hand, for the impostor user we can see that the median value is below 0.45. We have adjusted the trust model from [Bou12] by having 1 type of reward, and 2 different levels of penalty. We have tested our system with various parameter values in the trust model. The values of the parameters represented in this paper were the ones giving the results from Section 7. The used trust model algorithm is as follows:

1. Build the classifier model with the training data of both the genuine and the impostor users (see Section 3).
2. Start at 100% trust (so initially we have $C = 100$).
3. For each event in the test set of a specific user:
   (a) Determine from the classifier model what the probability P_g of the genuineness of this event is.
   (b) Increase or decrease the trust value C according to the probability P_g of the genuineness of the event:

$$C = \begin{cases} \min(C + P_g, 100) & \text{if} \quad P_g \geq 0.5 \\ \max(C - (1 - P_g), 0) & \text{if} \quad 0.3 \leq P_g < 0.5 \\ \max(C - 1, 0) & \text{if} \quad P_g < 0.3 \end{cases} \tag{1}$$

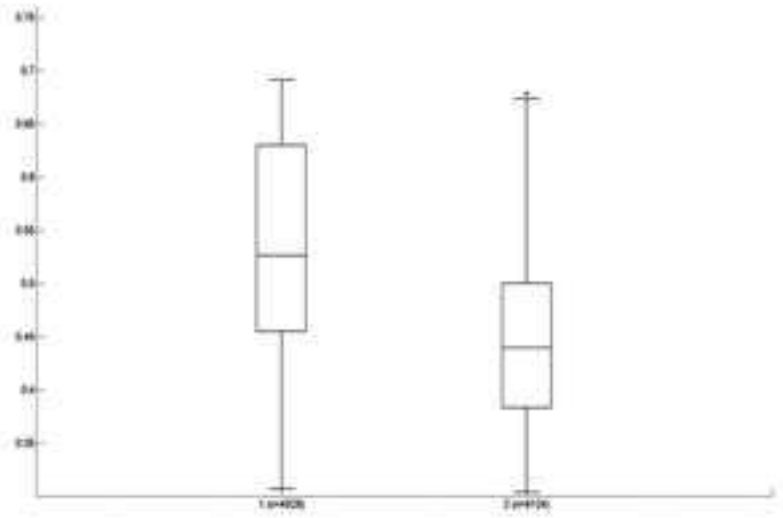

Figure 2: Events classification probability distribution for a genuine user (1) and impostor user (2).

4. Compare the trust value C with the lock out threshold Tr:
   (a) If $C \geq Tr$ then continue with step 3.
   (b) If $C < Tr$ then lock the system. During analysis this means a simulation of a new log on and continuation at step 2.

In our analysis we used both global thresholds and person specific thresholds. When testing the system we found that a global threshold was not giving good results, meaning that impostors were not detected. A personal threshold, based on the stability of the typing of the genuine person was also used in our research. The main goal was to have a personal threshold where the genuine user was never locked out of the system. This was achieved by selecting the threshold slightly below the least attained trust value when evaluating the test data of the genuine user. In a real system this cannot be done in this way of course. In that case will the system, after the training of the classifier model, first use some data to determine the personal threshold level for the specific user. Only after also the personal threshold has been set will the system of the user be protected with continuous authentication via mouse dynamics.

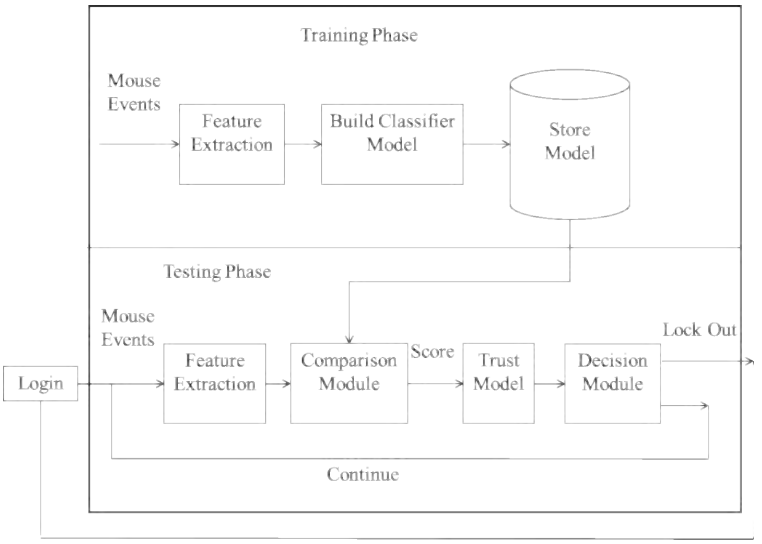

Figure 3: Block Diagram of the system.

6 Analysis method

In this section, we are going to discuss the methodology of our system. The system was divided into two basic phases (see Figure 3).

I. Training Phase: In the training phase, the training data (see Section 4.3) was used to build the classifier model and store the model in a database for use in testing phase. Each genuine user has his/her own classifier model. This means that we have build 49 different classifier models.

II. Testing Phase: In the testing phase, we are going to use test data which was separated from the training data for comparison. In the comparison, we will use the model stored in the database and obtain the classifier score (probability) on each sample of the test data. This score will then be used to update the trust value C in the trust model (see Section 5). Finally, the trust value C was used in the decision module, to determine if the user will be locked out or can continue using the PC. This decision was made based on the current trust value and the lockout threshold.

In the testing phase will the performance of the system be measured in terms of Average Number of Genuine Actions (ANGA) and Average Number of Impostor Actions (ANIA). In this case an action of the user can be anything done with the mouse, for example moving the mouse, clicking, or drag-and-drop. This is done by counting the number of test data samples of the genuine or an impostor user that can be used in the model from Section 5 before a user is locked out. Any user will always start at the trust value $C = 100$ and in

case of a trusted test data sample, i.e. the probability outputted by the classifier model is above 50%, this trust value will go up (with a maximum of 100). In case the probability from the classifier model is below 50% then the trust value C will decrease. This means that if the user's typing is in accordance with the classifier model, then the trust in the genuineness increases, otherwise it will decrease.

The data samples from the genuine user will in most cases get a high probability from the classifier and sometimes a low probability. This means that most often the trust will increase and sometimes it will decrease. This then results in a trust value that will remain at a high level. For impostor users this situation is the opposite. Often the trust value will decrease and sometimes, when the behaviour of the genuine user is mimicked correctly, the trust value will go up. The general trend for the trust value will however be downwards, and once the trust value reaches below the threshold, then the user will be locked out.

The average number of actions an impostor can do before being locked out will be the ANIA value, while the average number of actions for the genuine user will be denoted by ANGA. The goal is obviously to have ANGA as high as possible (in fact we try to never lock out a genuine user), while at the same time the ANIA value must be as low as possible. The last is obviously to assure that an impostor user can do as little as possible, hence he/she is detected as quick as possible. Given a fixed model of a user, then changing the parameters in the trust model of Section 5 will either increase both values of decrease both values.

7 Performance

In this section we are going to discuss the result we have achieved from this research and the discussion related to the result.

Although we used 6 different ML algorithms (see Section 3) to evaluate the system, we will report here only the results of the SVM classifier. As mentioned before were some impostors not detected for each of the other 5 classifiers, even when using a personal threshold. For the other 5 classifiers the probability that an impostor user was not detected when using a personal threshold ranged from 12% (for k-NN) to 76% (for Decision tree learning). Therefor we will focus only on the results obtained with SVM in this paper.

We created 49 SVM classifier models, i.e. one for each user. For the performance analysis we calculated the ANGA and ANIA values both for fixed lock out thresholds and for personal thresholds. We first tested our 49 models with 5 fixed lockout thresholds of 40, 60, 80, 85, and 90. We observed that the genuine users were never locked out because those users are very stable in their way of using the mouse with respect to the used model. In particular does this mean that the trust value of a genuine user in his own classifier model never drops below 90. In Table 1 the results are shown for these fixed thresholds. This table shows the average ANIA but also the number of impostors that are not detected by the system, given the particular settings.

From Table 1 we see that a lower fixed threshold means that an impostor can do more actions and more impostors will not be detected. Although this is a logical conclusion,

Threshold	ANIA	undetected
40	1363	190
60	999	135
80	583	45
85	385	28
90	192	6

Table 1: ANIA results for fixed lockout threshold

the table is intended to give an indication of how much more an attacker can do when the threshold is lowered. In particular we can see that lowering the threshold from 90 to 85 means more or less a doubling of the average number of actions an impostor can do, i.e. from 192 to 385. On top of that is the number of impostors that are not detected more than 4 times as high for threshold 85 compared to threshold 90.

Based on the results for the fixed threshold did we decide to use a user specific threshold. This user specific threshold was chosen in such a way that the genuine user was never locked out. More specifically did the personal thresholds range between 90.8 and 96.7. A user with a higher personal threshold will be more consistent in his own mouse usage and will have fewer (consecutive) penalties that decrease the trust value. The average ANIA value was in this case equal to 96 (with a standard deviation of 79), meaning that an impostor was, on average, detected after 96 mouse actions.

The value of ANIA depended highly on the specific genuine user. One of the users had an ANIA of only 9, i.e. on average an impostor was detected after only 9 mouse actions. On the other hand, in the given dataset, the worst performance was from a user where, based on his classification model, the value of ANIA was 344. It is obvious that there is a wide spread of ANIA values, meaning that not only the average ANIA value of 96 is important, but also the standard deviation of the ANIA values should be considered.

Figure 4 shows the trust value of a genuine user when tested against almost 5000 test data samples. Although it can be seen that the trust value goes down slightly, will this be corrected automatically because the genuine user uses the mouse in the correct manner, according to the classifier model. Figure 5 shows the trust value of an impostor user, and we see that generally the trust value will go down. After each lockout the trust value is reset to 100 again.

8 Conclusion and Future work

In this research, we have shown a new way of building a continuous authentication system where the system will decide the trust in the genuineness of the user in each and every event. We have tested this idea with 6 different ML algorithms in a publicly available mouse dynamics dataset. The genuine user has never been locked out throughout the session whereas the Average Number of Impostor Actions (ANIA) is 96 for the SVM clas-

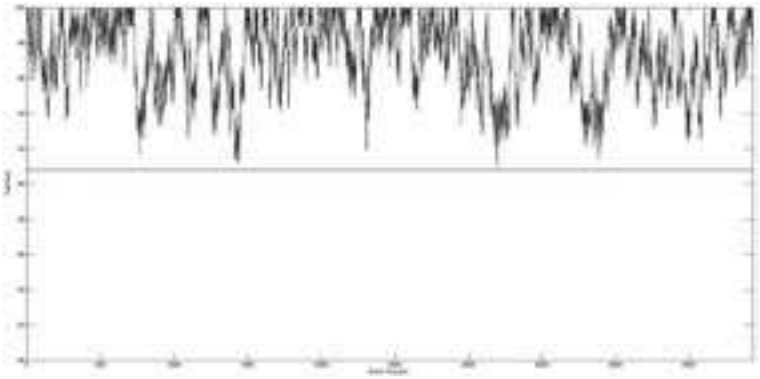

Figure 4: Trust value for genuine user tested with the genuine test set.

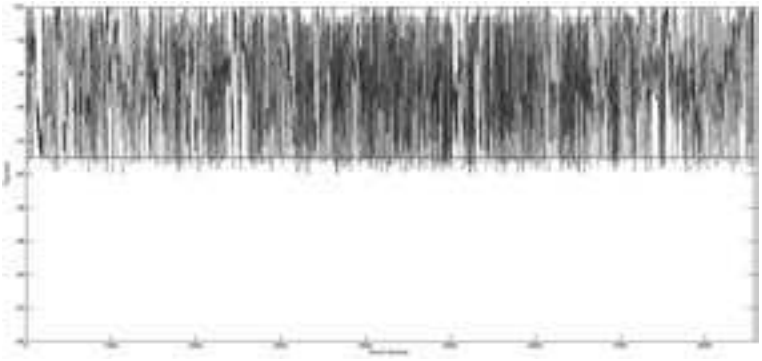

Figure 5: Trust value for genuine user tested with the impostor test set.

sification algorithm with the person based threshold. We have found that the performance of the user is very much dependent upon the stability of user's own mouse dynamics.

We stress here that the results are obtained given the specific trust change function from Equation 1. The boundaries for P_g or the changes in the trust value C used in this equation are taken as fixed, and we have not attempted to optimize these values. Even better results could have been reached, if these values are optimized, or even if these are not taken as global values, but also as person dependent. In fact we have done some tweaking of the trust function from Equation 1, where the value of 0.3 was replaced with 0.4 and in that case the average ANIA went down to 94 (with standard deviation of 78).

The results we have obtained from this research are satisfactory to validate this approach of continuous authentication using mouse dynamics, but we need more per event features to create a real world continuous authentication system which is out of scope for this dataset. Other examples of per event feature attributes can be the size of the active region where the mouse event occurs, the centroid of the active region and the relative position of the

mouse in that region, curvature, or duration between mouse button pressure and release. Future work will be to build our own dataset to prove our concept.

References

[AT07] A.A.E. Ahmed and I. Traore. A New Biometric Technology Based on Mouse Dynamics. *IEEE Transactions on Dependable and Secure Computing*, 4(3):165–179, 2007.

[Bou12] P. Bours. Continuous keystroke dynamics: A different perspective towards biometric evaluation. *Information Security Technical Report*, 17:36–43, 2012.

[CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011.

[EM03] R.A.J. Everitt and Peter W. McOwan. Java-based Internet biometric authentication system. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(9):1166–1172, 2003.

[FEM+12] C. Feher, Y. Elovici, R. Moskovitch, L. Rokach, and A. Schclar. User identity verification via mouse dynamics. In *Information Sciences*, 2012.

[GF03] H Gamboa and A. Fred. An identity authentication system based on human computer interaction behavior. In *Proc. of the 3rd Intl. Workshop on Pattern Recognition in Information Systems*, 2003.

[GF04] H. Gamboa and A. Fred. A behavioral biometric system based on human computer interaction. In *Proceedings of SPIE*, 2004.

[HFH+09] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[LCL12] Chien-Cheng Lin, Chin-Chun Chang, and Deron Liang. A New Non-intrusive Authentication Approach for Data Protection Based on Mouse Dynamics. *Biometrics and Security Technologies, International Symposium on*, 0:9–14, 2012.

[NTA10] Y. Nakkabi, I. Traoré, and A.A.E. Ahmed. Improving Mouse Dynamics Biometric Performance Using Variance Reduction via Extractors with Separate Features. *IEEE Transactions on Systems, Man And CyberneticsPart A: Systems and Humans*, 40:1345–1353, 2010.

[PB04] M. Pusara and C.E. Brodley. User Re-Authentication via Mouse Movements. In *ACM Workshop Visualization and Data Mining for Computer Security*, 2004.

[SCG12] Chao Shen, Zhongmin Cai, and Xiaohong Guan. Continuous authentication for mouse dynamics: A pattern-growth approach. In *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*, pages 1–12, 2012.

[Sch06] D.A. Schulz. Mouse Curve Biometrics. In *Proc. Biometrics Symp.: Special Session Research at the Biometric Consortium Conference*, 2006.

[ZPW11] N. Zheng, A. Paloski, and H. Wang. An Efficient User Verification System via Mouse Movements. In *18th ACM conference on Computer and communications security*, 2011.